Maschinelles Lernen zur Erkennung personenbezogener Daten in deutschsprachigen Textdokumenten

Learning to Detect Personal Information in German Text Documents

Bachelor-Thesis von Nils Thoma

29. Oktober 2018

TECHNISCHE UNIVERSITÄT DARMSTADT

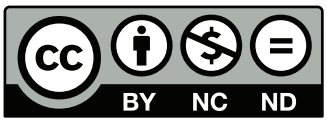
Fachbereich Informatik
Knowledge Engineering Group

Maschinelles Lernen zur Erkennung personenbezogener Daten in deutschsprachigen Textdokumenten
Learning to Detect Personal Information in German Text Documents

Vorgelegte Bachelor-Thesis von Nils Thoma

1. Gutachten: Prof. Dr. Johannes Fürnkranz
2. Gutachten: Msc. Markus Zopf

Tag der Einreichung: 30.10.2018



Erklärung zur Abschlussarbeit gemäß § 23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Nils Thoma, die vorliegende Bachelor-Thesis ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Fall eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung überein.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Datum / Date: 30.10.2018

Unterschrift / Signature:

Abstrakt / Kurzfassung

Die Analyse von großen Daten hat in den vergangenen Jahren bedeutend an Popularität gewonnen, besonders unter dem Stichwort 'Big Data'. Größere Rechenkapazitäten sowie die durch die massive Nutzung des Internets schnell wachsende Menge Daten haben diesen Trend beflügelt. Um Missbrauch vorzubeugen und persönliche Daten zu schützen, existieren Auflagen (in der Europäischen Union die DSGVO) welche die Verarbeitung sensibler Daten regulieren. Als Resultat dieser Regelungen ist es für manche Verarbeitungsschritte notwendig, personenbezogene Daten zu entfernen. Da Unternehmen ein Interesse daran haben, trotz dieser Reglungen Wissen aus den Daten gewinnen zu können, ist der Einsatz einer Anonymisierung gegenüber einer Löschung vorzuziehen. Denn so kann aus den Daten weiterhin ein Nutzen gezogen werden.

In der Industrie werden für die automatische Durchführung der Anonymisierung Systeme genutzt, welche auf klassischen Methoden wie Regulären Ausdrücken und Regeln basieren. Doch diese zeigen bisweilen unzufrieden stellende Ergebnisse, besonders bei unregulären Daten, wie es zum Beispiel bei Chat Verläufen aus dem Support eines Unternehmens der Fall ist. In dem eng mit der Anonymisierung verwandten Bereich der Named Entity Recognition (NER) hat sich der Einsatz von Systemen auf Basis Maschinellen Lernens (ML) als erfolgreich gezeigt.

Diese Arbeit geht der Frage nach, inwiefern sich verschiedene ML-Modelle aus der NER in den Bereich der Anonymisierung übertragen lassen und vergleicht ihre Leistungen gegenüber einem in der Industrie eingesetzten Anonymisierungssystem, welches auf klassischen Methoden basiert. Dafür werden verschiedene Tests auf regulären sowie auf unregulären Daten durchgeführt.

Für den Einsatz von ML-Systemen sind entsprechende Datensätze nötig, um sie trainieren und testen zu können. Da keine deutschen Korpusse im Bereich der Anonymisierung existieren, werden im Rahmen dieser Arbeit außerdem die Wiedervervollständigung eines anonymisierten Chat-Korpus (unreguläre Daten) sowie die Generierungen eines kleinen E-Mail Datensatzes mit diversen Anwendungsfällen aus dem Bereich des Kundensupports in Unternehmen (reguläre Daten) durchgeführt.

Anhand diverser Evaluationsmethodiken wird gezeigt, dass der Einsatz von ML-Modellen aus dem Bereich der NER zu guten Ergebnissen in der Anonymisierung führt. Dabei wird die Leistungen des Vergleichssystems aus der Industrie von allen ML-Ansätzen übertroffen. Besonders gute Ergebnisse erreichen Conditional Random Fields, sowie die Kombination eines Bidirektionalen Long-Short-Term-Memory Systems mit einem Convolutional Neural Network.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Diese Arbeit befasst sich mit der Anonymisierung unstrukturierter Texte deutscher Sprache. Der Fokus liegt dabei auf der Anonymisierung von Chat-Nachrichten sowie E-Mails. Die Durchführung soll mit Techniken des Maschinelles Lernens (ML) realisiert werden, die aus dem Bereich der Named Entity Recognition (NER) stammen. Dort erbringen ML-Systeme seit einiger Zeit (im Vergleich mit alternativen Modellen) die besten Leistungen (siehe Sektion 3.2.2). Auf diese Weise kann die Frage geklärt werden, inwieweit sich dies auf den Bereich der Anonymisierung übertragen lassen. Dazu werden die Leistungen der Systeme auch mit einem produktiv genutzten System (Sektion 3.1) verglichen, welches in der Industrie zum Einsatz kommt. Dieses basiert auf sogenannten klassischen Methoden, wie zum Beispiel regulären Ausdrücken, näheres dazu in Sektion 3.1.
Die Systeme werden dafür auf 2 verschiedenen Korpera trainiert und getestet. Bei einem Datensatz handelt es sich um einen, als Teil dieser Arbeit Wieder-Vervollständigten, anonymisierten Chat Korpus. Der andere Datensatz besteht aus einem Satz an generierten E-Mails, welcher diverse Anwendungsfälle aus dem Bereich des Kundensupports in Unternehmen enthält.

1.1 Motivation

Die Analyse von großen Mengen an Daten hat in den vergangenen Jahren bedeutend an Popularität gewonnen, besonders unter dem Stichwort 'Big Data' [2]. Subjekt dieser Analysen sind dabei diverse Arten von Daten, welche in Unternehmen sowie von Nutzern im Internet generiert werden. Die Menge dieser Daten, welche zur Analyse bereit stehen, hat sich dabei in den letzten Jahren durch die steigende Nutzung des Internets, sowie der Digitalisierung des privaten- sowie beruflichen Alltags durch Geräte wie Smartphones und IoT-Geräte, rasant vermehrt [2]. Auch die Rechnerkapazitäten, die zur Analyse solch großer Datenmengen bereit stehen, sind massiv gewachsen [38]. Doch durch die systematische Analyse entstehen auch große Gefahren für die Gesellschaft, sowie für jede Einzelperson, wie zum Beispiel Clarke in "Big data, big risks" darlegt [20]. Daher existieren Regularien, um den Missbrauch solcher Daten vorzubeugen - in der Europäischen Union und damit auch in Deutschland wurden diese Regularien mit der Einführung der Europäische Datenschutz-Grundverordnung (DSGVO) im Mai 2018 (mehr dazu in Sektion 2.1) verschärft. Dadurch ist es in vielen Fällen notwendig, personenbezogene Daten zu entfernen, bevor sie in weiteren Schritten verarbeitet werden dürfen (siehe Sektion 2.1.2). Doch Unternehmen haben, im Rahmen von 'Big Data', ein Interesse daran, aus diesen Daten trotzdem Wissen zu gewinnen. Daher ist der Einsatz einer Anonymisierung gegenüber einer Löschung vorzuziehen, denn so kann aus den Daten weiterhin ein Nutzen gezogen werden, zum Beispiel um Vorteile im Marketing zu erhalten [84].
Diese Anonymisierung wird, abgesehen vom medizinischen Bereich, in der Regel mit Klassischen Methoden (siehe 3.1) durchgeführt (siehe 3). Doch diese Systeme zeigen bisweilen keine zufrieden stellende Ergebnisse, besonders bei unregulären Daten. Solche Daten weisen gehäuft Rechtschreib-, sowie Grammatikfehler auf und entstehen zum Beispiel bei Chat Verläufen aus dem Support eines Unternehmens (mehr dazu in Sektion 4.4). Systeme aus der Anonymisierung im medizinischen Bereich lassen sich auf solchen Daten nur begrenzt einsetzen, da in diesem Bereich die Daten einer sehr regulären Struktur folgen und andere Begrifflichkeiten nutzen (siehe 3.2.1). Viele Parallelen weißt die Aufgabenstellung zu dem verwandten Bereich der NER auf, welcher sich mit der Erkennung von sogenannten Named Entities (NE), wie Namen oder Orte, in Texten befasst. Doch auch diese Systeme lassen sich nicht direkt auf eine Anonymisierung anwenden, da einige Differenzen bestehen (näheres in Sektion 2.2.3).
Als Folge einer steigenden Wichtigkeit der Datenanalyse, verknüpft mit der Notwendigkeit einer Anonymisierung, besteht also der Bedarf nach Systemen, welche eine Anonymisierung auch auf unregulären Daten mit guten Ergebnissen durchführen können. Denn nur so ist es oft möglich, die Daten in einem

[2] Mit Big Data werden im Allgemeinen große Datenmengen bezeichnet, welche "durch die „4 Vs“ Volume (Menge), Velocity (Geschwindigkeit der Mengenzunahme), Variability (Vielfalt bezüglich Inhalt, Quellen und Struktur) und Veracity (Verlässlichkeit oder auch Wahrhaftigkeit) gekennzeichnet sind" [84]

rechtlich korrekten Rahmen zu verarbeiten. In dem Bereich der Anonymisierung, sowie in verwandten Gebieten wie der NER, existieren gerade im Bereich der deutschen Sprache außerdem vergleichsweise wenige Arbeiten (mehr dazu in 3). Als Ansatz bieten sich ML-Systeme an, wie sie auch erfolgreich in dem verwandten Bereich der NER eingesetzt werden (siehe Sektion 3.2.2). Daher erforscht diese Arbeit, inwiefern sich ML-Systeme aus dem Bereich der NER in dem Bereich der Anonymisierung einsetzen lassen und ob sich die Differenzen zwischen den Bereichen, welche in Sektion 2.2.3 dargestellt sind, überwinden lassen.

1.2 Struktur der Arbeit

Im folgenden Kapitel wird das grundlegende Wissen aller relevanten Bereiche erläutert. Dies bildet die Grundlage, um dieser Arbeit folgen zu können. Begonnen wird dazu mit einer kurzen Einführungen in den Datenschutz. Dieser begründet zum einen die Notwendigkeit einer Anonymisierung, zum anderen bildet er auch die rechtliche Grundlage für die Durchführung einer solchen. Darauf aufbauend werden Methodiken des ML erläutert, welche für die Durchführung der Anonymisierung eingesetzt werden. Des weiteren werden Maße vorgestellt, mit deren Hilfe sich die Leistungen dieser Systeme evaluieren lassen. Unter Einsatz dieser Maße werden in dem darauf folgenden Kapitel verschiedene Methoden aus den Bereichen der Anonymisierung sowie der NER vorgestellt und verglichen. Ausgehend vom momentanen Stand der Technik wird zuerst das Vergleichssystem aus der Industrie vorgestellt, bevor auf alternative Ansätze mit Maschinellem Lernen eingegangen wird.
Anhand der daraus gewonnen Erkenntnisse werden die erfolgversprechendsten Modelle im nächsten Kapitel unter diversen Bedingungen getestet. Im Zuge dessen werden auch die dafür notwendigen Daten, sowie der Prozess deren Erstellung, näher betrachtet. Anschließend erfolgt erst eine allgemeine Evaluierung, bevor in einer grundlegenden Analyse tiefer gehende Erkenntnisse erarbeitet werden.
Abschließend wird das neu gewonnene Wissen in einem Fazit zusammengefasst, sowie über zukünftige Möglichkeiten der Forschung in diesem Bereich diskutiert.

2 Grundlagen

Im folgenden werden die Grundlagen aller für die Arbeit relevanten Bereiche erklärt. Zuerst wird im Bereich Datenschutz erarbeitet, welche rechtlichen Grundlagen Anonymisierungen notwendig machen und welche Arten von Daten daraus folgend anonymisiert werden müssen. Grundlegend für die Durchführung von Anonymisierungen ist die Fähigkeit, natürliche Sprache zu verarbeiten - daher werden daraufhin die wichtigsten Methodiken aus dem Bereich des Natural Language Processing (NLP) erläutert. Für die tatsächliche Durchführung der Anonymisierung werden Techniken des Maschinellen Lernens eingesetzt - in Sektion 2.3 werden alle benutzten Methodiken, wichtige Begrifflichkeiten sowie weitere relevante Grundlagen vermittelt. Da solche Modelle nicht fehlerfrei arbeiten, werden des weiteren noch Maße vorgestellt, mit deren Hilfe die Leistungen solcher Modelle beurteilt werden können.

2.1 Datenschutz

Die Anonymisierung von Texten wird durch den Datenschutz motiviert - denn dieser legt fest, welche Daten schützenswert sind und daher entfernt werden müssen, bevor Texte zur weiteren Analyse genutzt werden dürfen [66]. Um festzustellen, welche Daten anonymisiert werden müssen, ist es daher zunächst wichtig zu verstehen, was genau Datenschutz bedeutet und welche Daten durch ihn geschützt werden. Im Duden wird Datenschutz definiert als

> *"Schutz der Bürger[innen] vor Beeinträchtigungen ihrer Privatsphäre durch unbefugte Erhebung, Speicherung und Weitergabe von Daten, die ihre Person betreffen" [26]*

Demzufolge schützt der Datenschutz alle Daten, die die eigene Person betreffen und zwar gegen Erhebung, Speicherung sowie Weitergabe. Pommerening unterstützt dies durch seine Definition:

> *"Datenschutz ist der Schutz von Daten vor Missbrauch, unberechtigter Einsicht oder Verwendung, Änderung oder Verfälschung, aus welchen Motiven auch immer" [67]*

Eng verwoben ist der Datenschutz durch den Schutz der Privatsphäre mit dem Grundgesetz des 'allgemeinem Persönlichkeitsrechts' und besitzt damit auch einen Verfassungsrechtlichen Schutz [81].
In den Mitgliedsstaaten der Europäischen Union (und damit auch in Deutschland) übernimmt seit dem 25. Mai 2018 die Europäische Datenschutz-Grundverordnung (DSGVO) die Regelung des Datenschutzes - dementsprechend spielt sie als rechtliche Grundlage eine wichtige Rolle in den nächsten Sektionen dieses Abschnitts.

2.1.1 Zu schützenden Daten

Nachdem die Umrisse des Begriffes 'Datenschutz' geklärt wurden, stellt sich die Frage, welche Art von Daten es denn sind, die, wie der Duden es formuliert, "ihre Person betreffen" [26]. Die DSGVO bezeichnet solche Daten als 'personenbezogene Daten' und definiert sie wie folgt:

> *"'personenbezogene Daten' alle Informationen, die sich auf eine identifizierte oder identifizierbare natürliche Person (im Folgenden 'betroffene Person') beziehen; als identifizierbar wird eine natürliche Person angesehen, die direkt oder indirekt, insbesondere mittels Zuordnung zu einer Kennung wie einem Namen, zu einer Kennnummer, zu Standortdaten, zu einer Online-Kennung oder zu einem oder mehreren besonderen Merkmalen, die Ausdruck der physischen, physiologischen, genetischen, psychischen, wirtschaftlichen, kulturellen oder sozialen Identität dieser natürlichen Person sind, identifiziert werden kann" (DSGVO, Art 4)*

Eine grundlegende Voraussetzung dafür, dass Daten als personenbezogen gelten ist dementsprechend, dass die betroffenen natürliche Person identifizierbar ist. Die konkrete Ausprägung kann hierbei sehr divers sein, wie folgendes Beispiel zeigt: "Der Fahrer des grünen Porsches wurde gestern in Frankfurt geblitzt" - Dieser Satz beinhaltet auf den ersten Blick keine zu schützenden Daten, da die erwähnte Person

('Der Fahrer') erst einmal nicht identifizierbar ist und damit nach *(DSGVO, Art 4)* nicht in den Bereich der 'personenbezogene Daten' fällt. Liegen aber weitere Informationen, zum Beispiel aus dem Kontext des Satzes, vor, sodass der Fahrer identifizierbar wird, ist die Information, dass der Fahrer gestern geblitzt wurde, schützenswert. Auch andere Umstände können dies beeinflussen: Existiert nur ein Porsche, welcher im Raum Frankfurt fährt, führt dass auch dazu, dass die Informationen schützenswert sind. Die Bestimmung, ob eine erwähnte Person identifizierbar ist und die Daten damit schützenswert sind, benötigt also unter Umständen viel (potentiell unzugängliches) Wissen. Des weiteren besteht eine starke Abhängigkeit zum Kontext, in dem einzelne Wörter oder Sätze stehen - daher ist auch nicht zu erwarten, dass eines der automatisierten Systeme, wie sie in dieser Arbeit behandelt werden, solch etwas zu leisten vermag - schließlich besitzen sie keinen Zugriff auf solche Daten. Außerdem existiert aus diesen Gründen auch keine vollständige Liste aller Datenarten, die anonymisiert werden müssen. Im Rahmen dieser Arbeit orientieren wir uns daher zum einen an den Daten, die in den eingesetzten Korpera als 'zu anonymisieren' markiert wurden (siehe dazu Sektionen 4.1 sowie 4.2), sowie an den Daten, die das in Sektion 3.1 vorgestellte, in der Industrie eingesetzte Tool, per Spezifikation abdeckt. Damit ergibt sich folgende (unvollständige) Aufstellung an Daten, die anonymisiert werden müssen, falls die betreffende Person als identifizierbar gilt:

- Allgemeine Personendaten
  - Name
  - Geburtsdatum / Alter
  - Geschlecht
  - Herkunft
  - Beruf
  - Sexuelle Orientierung
  - Religion
  - Wohnort
  - Politische Orientierung
- Kontaktdaten (z.B. E-Mail Adresse, Telefonnummern)
- Kennnummern (z.B. Kundennummer, Vertragsnummer)
- Zahlungsdaten (z.B. Kreditkartennummer, Kontonummer)
- Onlinedaten (z.B. Passwörter, IPs, Gerätekennungen)

Da dies die Daten darstellt, welche entfernt werden müssen, um ein Dokument zu anonymisieren, werden sie im restlichen Verlauf der Arbeit auch als 'zu anonymisierende Daten' bezeichnet.

2.1.2 Anonymisierung und Pseudonymisierung von personenbezogenen Daten

Im Rahmen der Anonymisierung von Daten gibt es eine grundsätzliche Unterscheidung zwischen Pseudonymisierung sowie Anonymisierung. Im Rahmen einer Pseudonymisierung werden zu anonymisierende Daten durch ein Pseudonym (zum Beispiel ein Zahlencode oder eine zufällige Entität des selben Typs, wie zum Beispiel eine zufällige Adresse) ersetzt - dabei werden wiederholte Vorkommen einer selben Entität durch das selbe Pseudonym ersetzt. So ist es auch nach der Pseudonymisierung Entitäten zu identifizieren, welche in einem Zusammenhang stehen. Die Zuordnung, welcher Zahlencode zu welchem ursprünglichen Datum gehört, werden in einem zusätzlichen Dokument verwahrt - ist man im Besitz dieses Dokumentes, kann man die Ursprungsdaten wieder her stellen. Die DSGVO unterstützt diese Definition:

"'Pseudonymisierung' die Verarbeitung personenbezogener Daten in einer Weise, dass die personenbezogenen Daten ohne Hinzuziehung zusätzlicher Informationen nicht mehr einer spezifischen betroffenen Person zugeordnet werden können, sofern diese zusätzlichen Informationen gesondert aufbewahrt werden und technischen und organisatorischen Maßnahmen unterliegen, die gewährleisten, dass die personenbezogenen Daten nicht einer identifizierten oder identifizierbaren natürlichen Person zugewiesen werden" (DSGVO, Art 4)

Für die Anonymisierung bietet die DSGVO keine gesonderte Begriffsdefinition, sondern erwähnt anonymisierte Informationen nur als

"personenbezogene Daten, die in einer Weise anonymisiert worden sind, dass die betroffene Person nicht oder nicht mehr identifiziert werden kann" (DSGVO, EG 26)

Alternativ lässt sich aber eine Definition des Bundesdatenschutzgesetzes (BDSG) heranziehen, welches bis vor die Einführung der DSGVO seine Gültigkeit besaß:

"Anonymisieren ist das Verändern personenbezogener Daten derart, dass die Einzelangaben über persönliche oder sachliche Verhältnisse nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand an Zeit, Kosten und Arbeitskraft einer bestimmten oder bestimmbaren natürlichen Person zugeordnet werden können." (BDSG, §3)

Der wichtige Unterschied in der Definition zu der Pseudonymisierung stellt die Einschränkung "ohne Hinzuziehung zusätzlicher Informationen" dar, welche im Rahmen der Pseudonymisierung vorgenommen wird. Dies ist in keiner der beiden Definitionen der Anonymisierung enthalten. Eine Anonymisierung kann also aus einer Pseudonymisierung genommen werden, falls alle Informationen über die ursprüngliche Ausprägung der anonymisierten Entitäten restlos vernichtet worden sind. Die Einschränkung, welche das BDSG im Bezug auf die Wiederherstellung der Daten vornimmt ("nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand"), ist in der DSGVO nicht zu finden.
Es existieren mehrere Anwendungsfälle, in denen eine Pseudonymisierung oder eine Anonymisierung zum Einsatz kommen kann - exemplarisch werden hier zwei vorgestellt. So definiert die DSGVO den Grundsatz der Datenminimierung, nach welchem personenbezogene Daten "dem Zweck angemessen und erheblich sowie auf das für die Zwecke der Verarbeitung notwendige Maß beschränkt sein" *(DSGVO, Art 5)* müssen. Dementsprechend müssen zum Beispiel im Rahmen einer Analyse, für welche die genaue Ausprägung personenbezogener Daten irrelevant ist, diese möglichst entfernt werden. Für die Umsetzung wird in Artikel 25 explizit die Anwendung einer Pseudonymisierung vorgeschlagen.

*"[...] trifft der Verantwortliche sowohl zum Zeitpunkt der Festlegung der Mittel für die Verarbeitung als auch zum Zeitpunkt der eigentlichen Verarbeitung geeignete technische und organisatorische Maßnahmen – wie z. B. **Pseudonymisierung** –, die dafür ausgelegt sind, die Datenschutzgrundsätze wie etwa Datenminimierung wirksam umzusetzen [...]" (DSGVO, Art 25)*

Ein weiterer Anwendungsfall ergibt sich durch das Recht auf Löschung (DSGVO, Art 17), wonach der jeweilige Verantwortliche verpflichtet ist, "personenbezogene Daten unverzüglich zu löschen". Da nach einer erfolgten Anonymisierung per Definition keine personenbezogene Daten mehr vorhanden sind, wird in der Praxis statt einer Löschung häufig eine Anonymisierung durchgeführt.

2.2 Natural Language Processing

Natural Language Processing (NLP) umfasst ein Gebiet, welches sich mit der Verarbeitung von natürlicher Sprache befasst. Dies können sowohl Gespräche sein, als auch verschiedene schriftliche Texte aus dem Internet, Zeitungen oder ähnlichem [8]. NLP besitzt dabei enge Verbindungen zu ML: Viele NLP-Methodiken setzen ML ein und viele ML-Systeme, welche sich mit natürlicher Sprache befassen, setzen Methodiken aus dem Bereich des NLP ein, um Sprache effizient verarbeiten zu können (Vergleiche Sektion 3). Dementsprechend ist es auch essentiell für die Anonymisierung von Texten.

2.2.1 Grundlegende Techniken

Da natürliche Sprache zu den unstrukturierten Daten gehört (vergleiche Sektion 2.3.3), werden meist einige Techniken vor der weiteren Verarbeitung angewandt (auch Preprocessing genannt), um die Daten vorzubereiten und klarer zu strukturieren [8].

Part-of-Speech-Tagging

Part-of-Speech-Tagging (POS-Tagging) ist eine Sequence-Tagging-Aufgabe (näheres dazu in 2.3.1), bei der jedem Wort seine Wortart zugewiesen wird, wie zum Beispiel Nomen, Verben et cetera [8]. Häufig wird dies als Feature in NER eingesetzt (Vergleiche Sektion 3).

Tokenisierung

Tokenisierung im NLP beschreibt die Aufgabe, einen Eingabetext in vorgegebene Stücke (Tokens) aufzuteilen. Meist wird hierbei versucht, Wörter voneinander zu trennen, zum Beispiel anhand von Frei- oder Satzzeichen. So würden aus dem Satz ”Hallo, wie geht es dir?” die Tokens (”Hallo”; ”,”; ”wie”; ”geht”; ”es”; ”dir”; ”?”) entspringen [8]. Dieses Verfahren bietet sich vor allem dann an, wenn ein System auf der Basis einzelner Wörter arbeiten soll, wie zum Beispiel im Falle von Sequence-Tagging.

Stemming

Als Stemming im NLP bezeichnet man Verfahren, welche verschiedene Varianten eines Wortes durch ”abschneiden” einiger Buchstaben auf einen gemeinsamen Wortstamm abbilden. Diese Grundform wird entweder statt dem Wort, oder als zusätzliches Feature an die Systeme übergeben. Jedes Wort wird hierbei isoliert von seinem Kontext betrachtet. Zum Beispiel würden die Worte ”Kauf”, ”kaufe”, ”kaufen”, ”Käufer” alle auf den gemeinsamen Wortstamm ”kauf” abgebildet werden. Dies ist bei einem breiten Feld von Anwendungen hilfreich, da so eine geringere Anzahl an verschiedenen Wörtern untersucht werden muss und Verbindungen zwischen Wörtern mit einem gleichen Wortstamm geknüpft werden können [8]. Gleichzeitig kann es aber auch Probleme hervorrufen, da Informationen zum Beispiel über die Konjunktionen verloren gehen. Auch einige Wörter mit verschiedenen Bedeutungen werden aufgrund ihres Aufbaus auf den selben Stamm abgebildet. In der Regel überwiegen aber die Vorteile und ein besseres Ergebnis wird mit der Anwendung von Stemming erreicht [13].

Lemmatisierung

Der Prozess der Lemmatisierung ist eng verwandt mit dem des Stemming: Auch bei der Lemmatisierung ist es das Ziel, ein Wort auf dessen Grundform (genannt ’Lemma’) abzubilden - diese Grundform wird entweder statt dem Wort, oder als zusätzliches Feature an die Systeme übergeben. Die Lemmatisierung beschränkt sich, im Gegensatz zum Stemming, nicht auf das ”abschneiden” von Buchstaben, sondern wendet komplexere Systeme an, welche auch den Kontext eines Wortes miteinbeziehen. So wird zum Beispiel das Wort ”besser” auf ”gut” abgebildet, was der intuitiven Bedeutung gerecht wird - dies kann Stemming aufgrund des grundlegend unterschiedlichen Aufbaus der Wörter nicht erfassen und liefert dadurch in der Regel schlechtere Ergebnisse als die Lemmatisierung. Dafür ist die Lemmatisierung zum einen langsamer als Stemming, des weiteren sind die Systeme, welche sie durchführen, deutlich komplexer. Welches System verwendet werden sollte, hängt vom Anwendungsfall ab, die Lemmatisierung treffen ähnliche Vor- sowie Nachteile wie das Stemming [50].

Stopword-Filtering

Als Stoppwörter (Stopwords) werden im NLP häufig auftretende Wörter bezeichnet, welche in der Regel wenig Relevanz für den Inhalt eines Textes besitzen. Dazu zählen im Deutschen unter anderem (un-)bestimmte Artikel, Präpositionen, sowie Konjunktionen. Auch Satzzeichen werden hierbei oft dazu gezählt.
Beim Stopword-Filtering wird der Text in der Regel vorher durch Tokenisierung in Tokens umgewandelt

und anschließend werden mithilfe einer sogenannten Stopword-List (Liste aller Stoppwörter der jeweiligen Sprache) alle Stoppwörter herausgefiltert. Da Stoppwörter in der Regel kaum (für NLP-Systeme detektierbare) Relevanz für die Aussage eines Textes besitzen, erleichtert das Filtern die weitere Verarbeitung, da die Länge des Textes deutlich verkürzt wird [57]. Doch auch hier gilt, dass Informationen verloren gehen - so wird zum Beispiel sowohl ”Salz und Pfeffer” als auch ”Salz oder Pfeffer” auf ”Salz Pfeffer” abgebildet.

BIO-Kodierung

Mithilfe der BIO-Kodierung (auch IOB-Kodierung genannt) werden Labels für Sequenzen von Tokens, wie zum Beispiel in natürlicher Sprache, näher kodiert. So bezeichnet B den Beginn eines Labels ”B” (Begin), ”I” die Fortsetzung solch eines Labels (Inside) und ”O” (Outside) diejenigen Tokens, welchen kein gesondertes Label zugewiesen wurde. Dabei werden diese zusätzlichen Annotation (B, I, O) meist mit einem Bindestrich vor dem eigentlichen Label notiert, wie folgendes Beispiel aus dem Bereich der NER zeigt:

Tony	Stark	traf	Peter	am	Platz	der	Einheit
B-PER	I-PER	O	B-PER	O	B-LOC	I-LOC	I-LOC

Tabelle 1: Beispiel für den Einsatz einer BIO-Kodierung

Dieser Text wurde mithilfe von Tokenisierung in Tokens unterteilt und dann mit Labels der Kategorie Person (PER) sowie Ort (LOC) versehen. Im Allgemeinen wird in dem Bereich der NER fast ausschließlich auf Basis der BIO-Kodierung gearbeitet, auch in der Anonymisierung wird sie häufig eingesetzt (vergleiche Sektion 3).
Im Rahmen von NER wird die BIO-Kodierung sehr häufig eingesetzt (siehe Sektion 3.2.2), denn durch ihre Nutzung ergeben sich wichtige Vorteile. Zum einen können zwei direkt aufeinanderfolgende Entitäten des selben Typs korrekt getrennt werden, zum anderen ist es leichter zu erkennen, wenn eine Entität mit mehren Tokens nur teilweise erkannt wurde. Dazu zwei Beispiele, bei denen die Annotation ’N’ für einen Namen steht: ”Dabei waren: Peter Parker, Tony Edward Stark, ...” würde die Annotationen O, O, B-N, I-N, B-N, I-N, I-N erhalten und es wäre somit zu erkennen, dass es sich hierbei um 2 getrennte Entitäten handelt, im Gegensatz zu der Annotation ohne BIO-Notierung (O, O, N, N, N, N, N). Würde nun ein System nur den ersten sowie letzten Token von ”Tony Edward Stark” richtig erkennen (also N, O, N beziehungsweise B-N, O, B-N), wird ohne Einsatz der BIO-Kodierung nicht klar, ob das System einfach nur eine von drei Entitäten nicht erkannt hat, oder ob es sich dabei um eine Entität mit mehren Tokens gehandelt hat. Mithilfe der BIO-Kodierung kann der Fehler mit einem Abgleich der richtigen Annotation (B-N, I-N, I-N) richtig eingeordnet werden. In der Anonymisierung haben diese Vorteile nur eine bedingte Geltung: Denn das eigentliche Ziel der Anonymisierung ist es, möglichst viele Wörter, die zu anonymisieren sind, richtig zu erkennen. Doch eine richtige Kategorisierung und Trennung der Entitäten, welche sich im Ausgabetext widerspiegeln (Zum Beispiel ’{Name} wohnt in {Straße} {Stadt}’ im Gegensatz zu ’{?} wohnt in {?} {?}’), erhält mehr Informationen und erhöht die Lesbarkeit - so ist er für viele Analysen wertvoller. Daher wird auch diese Möglichkeit im Rahmen der Arbeit genutzt.
Erweiterungen der BIO-Kodierung, wie zum Beispiel BIOES, welche zusätzlich die Kategorien E (End) sowie S (Single) enthalten, werden hier nicht verwendet, da die meisten verwandten Arbeiten nur BIO-Kodierung nutzen (vergleiche Sektion 3).

2.2.2 Kodierungen von Wörtern und Sätzen

Natürliche Sprache kann meist nicht ohne weiteres an verarbeitende Systeme (zum Beispiel ML-Systeme zur Anonymisierung) übergeben werden: Ohne weitere Kodierung wäre es nur möglich, Buchstabe für Buchstabe in einer bekannten Textkodierung (ASCII, UTF-8) zu übergeben. Dies ist aber keine aussagekräftige Form, Wörter zu kodieren - sprachliche Konzepte sind auf Buchstabenebene so schwer für Maschinen zu erfassen, es werden auch keine Relationen zwischen ähnlichen Wörtern erfasst und selbst

kurze Wörter benötigen viele Eingabeneuronen (in neuronalen Netzen, vergleiche Sektion 2.3.7), um verarbeitet zu werden. Daher setzt man weitere Kodierungen auf Wort,-sowie Satzebene ein [58].

One-Hot Kodierung

Eine simple Kodierung ist die One-Hot Kodierung: Im NLP setzt diese Kodierung für einen Text mit n unterschiedlichen Wörtern (Vokabeln) Vektoren der Länge n ein. Dafür werden alle Vokabeln durchnummeriert - der Kodierungsvektor jedes Wortes ist nun ein Vektor bestehend aus Nullen - einzig der Index der zu der Vokabel gehörenden Nummer ist Eins. Die konkrete Anordnung der Wörter in dem Vokabular ist dabei irrelevant. Arbeitet man zum Beispiel auf einem Vokabular mit den Wörtern ['Peter', 'Bauer', 'Mein', 'Name', 'ist'], würde der Satz "Mein Name ist Peter Bauer" als '00100 00010 00001 10000 01000' kodiert werden [30].
Diese Kodierung ist sehr einfach zu berechnen und kann jedes Wort individuell kodieren - gleichzeitig werden die Kodierungs-Vektoren aber bei einem großen Text mit einem entsprechend umfangreichen Vokabular schnell sehr groß. So besitzt alleine die deutsche Gegenwartssprache einen Wortschatz von mehr als 300.000 Grundformen - selbst für eine Kodierung, welche nur Grundformen berücksichtigt, würde also jedes Wort durch einen Vektor der Länge 300.000 repräsentiert werden [27]. So besäße die Kodierung des Satzes von oben bereits eine Länge von 1,5 Millionen binären Werten. Des weiteren sagt die Kodierung nichts über den Zusammenhang zwischen den Wörtern aus, da die Wahl der Vektoren in der Regel rein abhängig von der (zufälligen) Reihenfolge im Vokabular erfolgt. Die Probleme der schnell wachsenden Vektoren können grundsätzlich unter dem Einsatz vom Stemming oder Lemmatisierung verringert werden (statt der Wörter werden ihre Wortstämme kodiert), aber nur bis zu einem gewissen Grad. Daher werden Embeddings für Eingabedaten häufiger eingesetzt [30]. Die One-Hot Kodierung wird hingegen häufig für die Kodierung von Labels im Falle einer Multiklassen-Klassifizierung (Sektion 2.3.1) genutzt.

Embeddings

Embeddings beschreiben das Konzept, jedem Wort (Word-Embeddings), beziehungsweise jedem Satz (Sentence-Embeddings), einen realwertigen Vektor zuzuweisen - diese haben typischerweise zwischen 50 und 300 Dimensionen [65]. Dies hat zum einen den Vorteil, dass es so durch die vergleichsweise kurzen Vektoren bei einem großen Vokabular relativ effizient ist, zum Beispiel einem ML-System natürliche Sprache als Input beziehungsweise Feature zu liefern - eine One-Hot Kodierung würde hierfür deutlich größere Vektoren benötigen. Das Beispiel aus dem obigen Paragraphen ("Mein Name ist Peter Bauer") könnte also durch Embeddings mit einer Länge zwischen 250 und 1500 (Fließkommazahlen) kodiert werden. Zum anderen werden diese Vektoren keinesfalls zufällig gewählt, sondern so, dass sie wertvolle Informationen beinhalten (mehr dazu weiter unten) - Wörter, welche ähnliche Bedeutungen besitzen, liegen im Vektorraum (auch Embedding-Space genannt) näher beieinander als Wörter, welche eine unterschiedliche Bedeutung haben [59]. Auch sind sie so angeordnet, dass sich mathematische Operationen anwenden lassen, wie Beispiele der sogenannten 'German Word Embeddings' [3] zeigen: Das Ergebnis aller mathematischen Operationen ist der naheliegendste Vektor für die Operation auf der linken Seite (gemessen in Kosinus Similarität [4]):

$$\begin{aligned} Frau + Kind &= Mutter \\ Frau + Hochzeit &= Ehefrau \\ Obama - USA + Russland &= Putin \\ Verwaltungsgebaeude + Buecher &= Bibliothek \\ Verwaltungsgebaeude + Buergermeister &= Rathaus \end{aligned} \tag{1}$$

Dadurch sind Embeddings sehr mächtig, denn viele sprachliche Konzepte werden in ihnen in einer kompakten, für Maschinen leicht verständlichen Form abgebildet - auch sprachenübergreifende Analysen

[3] Deutsche Word Embeddings, `https://devmount.github.io/GermanWordEmbeddings/`
[4] Die Kosinus Similarität (Cosine Similarity) ist ein Maß für die Ähnlichkeit zweier Vektoren

werden durch sie deutlich vereinfacht (Mithilfe von Multilingualen Embeddings, welche ähnlichen Wörtern über Sprachen hinweg ähnliche Vektoren zuweisen) [59].
Hierbei benötigen Embeddings in der Regel weder Stemming noch Lemmatisierung, sondern haben für jede Wortform einen eigenen Vektor. Dabei werden verschiedene Formen des selben Wortes oft nahe zueinander im Embedding-Space platziert.
Die Erstellung von Embeddings erfolgt meist unter Zuhilfenahme spezieller neuronaler Netze (Sektion 2.3.7) - ein häufig genutztes Modell ist dabei 'Word2Vec', welches von Mikolov et al. entwickelt und in mehreren Papern vorgestellt wurde [58] [60] [61]. Während es möglich ist, diese Modelle selbst zu trainieren, zum Beispiel für einen spezifischen Anwendungsfall, werden häufig bereits trainierte und evaluierte Modelle verwendet. Ein Beispiel für die Englische Sprache sind hierbei die GloVe Vektoren der Universität von Stanford, in der Deutschen Sprache werden häufig die oben erwähnten German Word Embeddings verwendet - diese werden auch im Rahmen dieser Arbeit eingesetzt. Diese wurden im Rahmen einer Bachelorarbeit auf deutschen Wikipedia-Artikeln mit insgesamt 651 Millionen Wörtern erstellt [24].

2.2.3 Named Entity Recognition

Named Entity Recognition (NER) beschäftigt sich mit der Erkennung von sogenannte Named Entities (NEs) in natürlicher Sprache, wie zum Beispiel von Personen oder Unternehmen [8] [82]. Es kann dabei wie Part-of-Speech-Tagging als Sequence-Tagging-Aufgabe (siehe Sektion 2.3.1) gehandhabt werden, nur dass hier statt der Wortarten die Art ihrer Bedeutung gelabelt wird, wie zum Beispiel 'Ort' oder 'Organisation' - eine Übersicht der häufigst genutzten Kategorien ist in Tabelle 2 gegeben, ein Beispiel ist im Abschnitt 'BIO-Kodierung' in Tabelle 1 zu sehen.

Named Entity	**Beispiele**
Person	Peter Parker; Kanzlerin Merkel
Organisation	Studierendenwerk Darmstadt; Muster GmbH
Ort	Karolinenplatz 5, Darmstadt; Herrngarten
Andere (MISC)	Berufe; Daten etc.

Tabelle 2: Häufig genutzte Kategorien für Named Entities (Orientiert an Sang et al. [82]). Andere Definitionen, wie zum Beispiel von Bird et al., definieren feinere Strukturen für die Kategorien 'Organisation', 'Ort' sowie 'Andere' [8]

Zu dem Bereich der Anonymisierung und Pseudonymisierung von personenbezogenen Daten besitzt es viele Parallelen: In beiden Anwendungsfällen erfolgt die Klassifizierung meist pro Wort (häufig als Sequence-Tagging), außerdem existieren viele Überschneidungen in der Art der zu detektierenden Entitäten (zum Beispiel müssen in beiden Anwendungsfälle Personen detektiert werden). Es existieren aber drei grundlegende Unterschiede:

1. Im Rahmen einer Anonymisierung bestimmt der Kontext, in welchem sich eine Entität befindet, mit, ob es erkannt werden muss oder nicht. So muss zum Beispiel das Datum in "Peter Parker wurde am 17.05.1995 geboren" anonymisiert werden, in "Am 14. März 2018 wurde Angela Merkel zur Bundeskanzlerin gewählt" hingegen nicht. Im Rahmen von NER spielt dies keine Rolle.

2. Während, wie beschrieben, ein Großteil der NEs auch anonymisiert werden muss, existieren auch einige NEs, welche nicht anonymisiert werden müssen. So wäre im obigen Beispiel der Name 'Peter Parker' zu anonymisieren, 'Angela Merkel', als Person des öffentlichen Lebens, hingegen nicht. Beide wären hingegen im Rahmen einer NER zu detektieren.

3. Einige Entitäten, welche anonymisiert werden müssen, fallen in keine der NE-Kategorien, wie zum Beispiel E-Mail-Adressen.

Diese Unterschiede gilt es im Rahmen der Experimente zu überwinden. Denn alle verwendeten Systeme, mit Ausnahme des industriellen Vergleichssystems, stammen aufgrund der großen Parallelen aus der NER (vergleiche Sektion 4.3).

2.3 Maschinelles Lernen

Maschinelles Lernen (ML) ist ein Konzept, welches sich grundlegend vom klassischen Ansatz, wie Computer programmiert werden, unterscheidet und in dieser Arbeit eingesetzt wird, um Daten zu anonymisieren. Während man bei einem klassischen Ansatz dem Computer Daten sowie konkrete Befehle (einen Algorithmus) übergibt, nach denen er die gewünschten Ergebnisse berechnen soll, übergibt man einem ML-System (oft auch als 'Lerner' bezeichnet) statt dem Programm die gewünschten Ergebnisse für die gegebenen Daten. Das ML-System versucht die Zusammenhänge zwischen den Daten und den gewünschten Ergebnissen zu erkennen beziehungsweise zu erlernen, um als Ergebnis ein sogenanntes Modell zu liefern. Dieser Prozess geschieht meist in mehreren Schritten - ein Modell wird erstellt, es wird festgestellt wie gut es ist und dann in einem nächsten Schritt verbessert. Diese Abfolge wiederholt sich, bis das Ergebnis zufriedenstellend ist (näheres dazu in Sektion 2.3.10). Das Modell, welches am Ende dieses Prozesses steht, verhält sich (im Idealfall) dann genau so, wie sich ein klassisches Modell mit dem entsprechenden Programm verhalten würde. Diesen Prozess bezeichnet man auch als das Trainieren des Modells (Dargestellt in Abbildung 1).

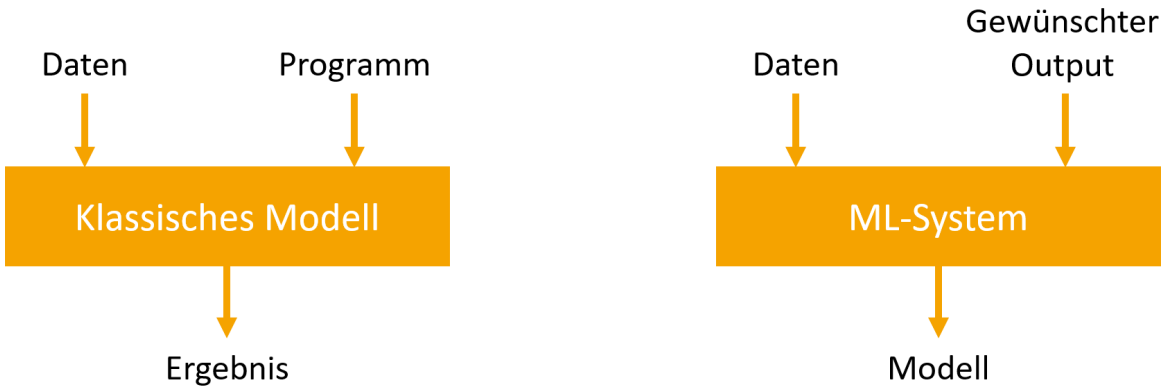


Abbildung 1: Berechnungsmodelle im Vergleich

Ist ein Modell einmal trainiert, kann es anhand einer gegebenen Eingabe ein Ergebnis, beziehungsweise eine Vorhersage, berechnen - ganz so, wie man ein klassisches Programm verwenden würde. Der Vorteil hierbei ist, dass es somit entfällt, eben solch ein Programm zu entwickeln. Außerdem können ML Systeme in der Lage sein, für komplexe Problemstellungen bessere Ergebnisse als manuell programmierte Systeme und als Menschen zu liefern. Ein bekanntes Beispiel hierfür ist Google's AlphaGo, welches mithilfe von Neuronalen Netzen (Sektion 2.3.7) den Weltmeister in dem Brettspiel GO [5] besiegt hat [76]. Dafür ist es aber nötig, das ML-System für die entsprechende Aufgabe zu konfigurieren (Sektion 2.3.2) und das richtige Konzept zu wählen. Außerdem werden Daten benötigt, mit denen das Training durchführen kann. Mehr dazu in Sektion 2.3.3.

Abbildung 2: Nutzung eines ML-Modells zur Gewinnung von Ergebnissen

[5] Go ist ein strategisches Brettspiel, welchem durch eine höhere Menge an möglichen Zügen eine höhere Komplexität als Schach zugesagt wird [76].

2.3.1 Klassifikation

Grundlegend existieren mehrere Formen von Aufgabenstellungen, welche mit ML behandelt werden können. So können zum Beispiel mithilfe von Regression unterliegende, mathematische Funktionen aus Datenpunkten hergeleitet werden, oder mithilfe von Clustering Gruppen (Cluster) in Datensätzen gefunden werden - dies kann zum Beispiel eingesetzt werden, um eine große Menge von Bildern in Gruppen einzuteilen, sodass zum Beispiel alle Bilder mit Hunden der selben Gruppe zugewiesen werden. Diese Arbeit hingegen beschäftigt sich mit der Aufgabe der Klassifikation von Datenpunkten - jedem Datenpunkt wird ein sogenanntes 'Label' (mehr dazu in Sektion 2.3.3) zugewiesen, welches seine Zugehörigkeit zu einer bestimmten Klasse ausweist. Dabei ist die endliche Menge der möglichen Klassen pro Aufgabenstellung festzusetzen (zum Beispiel auf 'Name' und 'Kein Name'). Unterscheidet man mehr als 2 Klassen, spricht man von einer Multiklassen-Klassifikation [9]. Den Fall, dass ein Datenpunkt mehr als einer Klasse zugehörig ist, wird im Rahmen dieser Arbeit nicht behandelt, da solch ein Fall sowohl im Bereich der Anonymisierung und Pseudonymisierung von personenbezogenen Daten als auch in der NER nicht relevant ist.

Sequence-Tagging

Sequence-Tagging, auch Sequence-Labeling genannt, ist eine besondere Art der Klassifikation, welche sich mit der Einteilung von Elementen einer Sequenz in Klassen beschäftigt. Hierbei wird jedem Element einer Sequenz ein 'Label' zugewiesen (in diesem Kontext auch 'Tag' genannt), zum Beispiel in einer Folge von Messwerten einer Maschine, wo Ausreißer entsprechend markiert werden. Solche Aufgabenstellung finden sich besonders häufig im Bereich des NLP sowie der NER, in welchen natürliche Sprache als Sequenz von Wörtern betrachtet wird [34]. Ebenso wird bei der Anonymisierung von Texten verfahren - hierbei wird jedem Wort entweder das Label 0 (muss nicht anonymisiert werden) oder das Label 1 (muss anonymisiert werden) zugewiesen. Erweiterungen davon ersetzen das Label 1 durch eine Auswahl von Klassen, die auch den Typus der zu anonymisierenden Entität wiedergeben, wie zum Beispiel 'Adresse' oder 'Name' (Multiklassenfall). Auf diese Weise wird es auch im Bereich der NER (siehe 2.2.3) und in dieser Arbeit eingesetzt.

2.3.2 Hyperparameter

Viele Modelle haben Steuerwerte, die den Lernprozess beeinflussen und nicht automatisch gelernt werden - dementsprechend müssen diese vom Entwickler gesetzt werden. Diese Parameter nennt man Hyperparameter - ein Beispiel für einen Hyperparameter wäre die Anzahl an Gruppen, die man für manche Clustering-Techniken vorgeben muss. Meist werden diese Parameter mehrfach während des Entwicklungsprozesses angepasst, um die jeweils optimalen Werte zu finden - diesen Vorgang nennt man Hyperparameter-Optimierung beziehungsweise Tuning [11]. Beispiele für Hyperparameter finden sich zum Beispiel in der Sektion 2.3.7 zu neuronalen Netzwerken.

2.3.3 Daten

Grundlegend für den Erfolg eines ML-Ansatzes sind die Daten, die zur Verfügung stehen, um ihn zu trainieren und ein Modell zu erstellen - nur wenn ausreichend Daten genügender Qualität vorhanden sind, hat das Modell die Möglichkeit, das Zielkonzept zu erlernen [63]. In dieser Sektion wird die grundlegende Terminologie im Bezug auf Daten im ML erklärt, sowie die Unterteilungen, welche in der Regel vorgenommen werden.

Labeling

Man unterscheidet 2 grundlegend unterschiedliche Gruppen von Daten: Daten, für welche das gewünschte Ergebnis (Label genannt) bekannt ist (gelabelte Daten), sowie Daten für die dieses Ergebnis unbekannt ist (ungelabelte Daten).
Ein Beispiel: Das Ziel ist es, mithilfe von ML Bilder zu unterscheiden, auf denen entweder ein Hund oder

eine Katze abgebildet ist. Eine Sammlung, in der nur Hunde beziehungsweise Katzenbilder enthalten sind, bezeichnet man als ungelabelte Daten. Besitzt man nun zu jedem Bild zusätzlich einen Eintrag in einer Datei, worin vermerkt ist, ob auf den jeweiligen Bildern ein Hund beziehungsweise eine Katze abgebildet ist (das Label), sind diese Daten gelabelt.
Es gibt Ansätze, welche nur mit gelabelten Daten umgehen können, aber auch welche, die aus ungelabelten Daten lernen können (mehr dazu in Sektion 2.3.5). Im Rahmen dieser Arbeit werden ausschließlich gelabelte Daten verwendet (näheres in Sektion 4.3)

Strukturierte und Unstrukturierte Daten

Im Rahmen von Daten im Bereich des maschinellen Lernens wird des weiteren zwischen strukturierten sowie unstrukturierten Daten unterschieden. Jeder Datenpunkt innerhalb eines strukturierten Datensatzes folgt einem bestimmten Schema - sie alle besitzen für bestimmte Attribute Werte aus einer vorgegebenen Menge von Optionen. Diese Optionen können sowohl relativ strikt (zum Beispiel ganzzahlige Werte $>= 0$ für ein Alter), als auch relativ weitreichend angegeben sein (zum Beispiel die Menge aller reellen Zahlen) - diese Werte müssen aber von sogenannter 'simpler' Natur sein, also keine komplexen Informationen beinhalten, die nur durch tiefer gehende Analyse extrahiert werden können, wie es zum Beispiel bei einem längeren Text der Fall ist (kurze Texte wie zum Beispiel Namen sind hingegen von simpler Natur und somit 'erlaubt') [1]. Strukturierte Daten lassen sich grundsätzlich in Tabellenform wiedergeben, wie in folgendem Beispiel:

Name	Geburtsdatum	Kontostand
Hannelore Musterfrau	13.10.1995	2.187,65
Max Mustermann	26.01.1964	-196,43

Tabelle 3: Beispiel für eine Tabelle mit strukturierten Daten

Unstrukturierte Daten können hingegen grundsätzlich alles sein: Freitext, Sprache, Audio, ... - die Daten müssen keinem festen Schema folgen - Blumberg et al. definieren sie im Bezug auf Datenbanken als "data that can't be stored in rows and columns" [10]. Ein Beispiel hierfür sind normales Textdokumente, welche keiner festen Struktur unterliegen.
Eine Zwischenform bilden die sogenannten 'semi-strukturierten Daten'. Diese bestehen im Kern aus unstrukturierten Daten, zu denen zusätzliche, strukturierte Informationen vorliegen, sogenannte Meta-Daten. Ein Beispiel dafür ist eine E-Mail, welche im Kern (der Text) unstrukturiert ist, aber strukturierte Meta-Daten besitzt (Absender, Empfänger, Datum, Uhrzeit, ...).
Im Allgemeinen sind strukturierte Daten einfacher für ein Computersystem zu verarbeiten, da die Informationen für das System direkt zugänglich sind. Für die Verarbeitung von unstrukturierten, beziehungsweise semi-strukturierten, Daten sind in der Regel als hingegen tiefer gehende Analysen als Zwischenschritt erforderlich [1].
Diese Arbeit beschäftigt sich mit der Verarbeitung von Sprache - daher handelt es sich bei den betrachteten Daten um unstrukturierte sowie semi-strukturierten Daten. Des weiteren verwendet diese Arbeit die Bezeichnung der 'unregulären' Daten - dies bezeichnet unstrukturierte Daten, welche keiner textlichen Struktur folgen, wie es zum Beispiel bei einer E-Mail der Fall ist (Begrüßung mit Anrede, Hauptteil, Verabschiedung) - solche bezeichnet man als 'reguläre' Daten. Außerdem zeichnen sich solche Daten durch eine stark erhöhte Häufigkeit von Rechtschreib- sowie Grammatikfehlern aus. Beispiele für solche Daten sind die Chat-Verläufe aus dem Dortmunder Chat Korpus, wie sie in dieser Arbeit großer Bestandteil sind - mehr dazu ist in Sektion 4.1 zu finden.

Datensätze

Hat das Training eines ML-Modells abgeschlossen ist es notwendig, dessen Ergebnisse zu überprüfen - ein Maß dafür zu bekommen, wie gut das Modell arbeitet. Nur so ist es möglich festzustellen, ob das Modell das Zielkonzept richtig erfasst hat - ähnlich wie Tests in der Softwareentwicklung durchgeführt

werden. Hierfür benötigt man dementsprechend gelabelte Daten - schließlich ist es notwendig, die erhaltene Vorhersage des Modells mit dem erwünschten Ergebnis abzugleichen. Hierbei ist es aber unbedingt notwendig Daten zu verwenden, welche nicht zum Training des Modells eingesetzt wurden - unter Verwendung bekannter Daten kann keine fundierte Aussage darüber getroffen werden, wie sich das System bei ihm unbekannten Daten verhalten wird. An einem Beispiel erläutert: Man könnte ein ML-System bauen, welches sich während des Trainings alle Bilder Pixelgenau mitsamt ihrem Label abspeichert und unter Testkonditionen dieses Label als Vorhersage wiedergibt, sollte ein Bild mit den gleichen Pixelwerten eingegeben werden. Testet man dieses System nun auf den Daten, mit denen es trainiert wurde, werden alle Ergebnisse richtig sein - da das System diese Bilder sowie ihre Label genau kennt. Versucht man aber Vorhersagen für Bilder zu erhalten, welche das System nicht kennt, kann es nichts besseres tun, als zu raten - denn es kennt diese neuen Bilder ja nicht. Somit hätte man sich ein System geschaffen, welches unter Testbedingungen vollkommen korrekte Ergebnisse liefert, aber unter Anwendungsbedingungen keinen Mehrwert liefert. Selbstverständlich ist dies ein überspitztes Beispiel, doch es stellt das konzeptionelle Problem dahinter dar. Daher trennt man im maschinellen Lernen die Datensätze grundlegend auf in:

Trainingsset Dies wird verwendet um das Modell zu erstellen und zu trainieren (Diese Daten können im Falle von unüberwachtem Lernen (vergleiche 2.3.5) ungelabelt sein).

Entwicklungsset Das Entwicklungsset wird verwendet, um das Modell während der Entwicklung zu testen und anpassen (zum Beispiel die Hyperparameter) zu können - Es ist wichtig dass hierfür nicht das Testset verwendet wird, denn auch dies könnte die Ergebnisse verfälschen.

Testset Dies wird verwendet, um das Modell abschließend zu testen - Erkenntnisse aus diesem Datensatz sollten nicht in das Modell einfließen, um eine Verfälschung des Ergebnisses zu vermeiden.

In welchen Proportionen diese Daten aufgeteilt werden, liegt im Ermessen des Entwicklers und muss an die Gegebenheiten angepasst werden - grundsätzlich wird aber ein Großteil der Daten zum Training verwendet, damit das Modell das Zielkonzept bestmöglich erlernen kann. Abhängig von der Größe des Datensatzes (umso größer der Datensatz, umso kleiner können Test-und Entwicklungsset anteilig sein), sollten Test-und Entwicklungsset jeweils zwischen 1% bzw. 15% des Gesamtdatensatzes ausmachen [8]. In der Regel sollten alle Datensätze repräsentativ für den gesamten Datensatz sein und eine ähnliche Verteilung der Beispiele aufweisen. In dem obigen Beispiel gesprochen: Wenn 40% aller Bilder Katzenbilder sind und 30% aller Bilder vor einem dunklen Hintergrund aufgenommen worden sind, sollten diese Verhältnisse in allen Datensätzen bestmöglich erhalten bleiben. Eine Ausnahme hiervon kann das Trainingsset sein, falls Oversampling angewandt wird (siehe Sektion 2.3.6).

Ausreißer und Overfitting

Ausreißer, auch Outlier genannt, sind Datenpunkte, die sich in ihren Werten deutlich von ihren sonst ähnlichen Datenpunkten im Datensatz unterscheiden. Dies kann zu Problemen für ML-Systeme führen: Auch wenn ein Großteil der Datenpunkte einem bestimmten Muster folgt, ist es möglich dass das ML-System dies nicht richtig erlernen kann, wenn es sich zu stark auf den einzelnen Ausreißer konzentriert. Ein Beispiel ist in Abbildung 3 zu sehen: Ein Großteil der Punkte der Klassen C0 sowie C1 folgen einem klaren Muster und ein Klassifikator kann in der Regel problemlos die grün eingezeichnete Linie finden, um beide Klassen zu trennen. Einzig der blaue Punkt bei den Koordinaten (7.2, 1.8) stellt einen Ausreißer dar: Würde ein Klassifizierer nun versuchen, auch diesen einzelnen Punkt richtig zu klassifizieren, würde er entweder scheitern, oder eine zu komplexe Lösung finden (die gelbe Linie als Erweiterung der grünen). Diese Lösung wird vielleicht den hier gezeigten Trainingsdaten gerecht, doch wird sie schlechter generalisieren als die Unterteilung mit der grünen Linie, sprich: Auf den Testdaten, in denen Punkte enthalten sind, die der Klassifizierer noch nicht gesehen hat, wird die gelbe Lösung mehr Fehler machen als die grüne. Denn zum Beispiel ein Punkt, der an (7.0, 2.3) erscheint, gehört höchstwahrscheinlich zur Klasse C1, würde von gelb aber als C2 klassifiziert werden. Diese zu starke Anpassung an Trainingsdaten,

die eine schlechte Fähigkeit zur Generalisierung zur Folge hat, nennt man Overfitting und tritt nicht nur im Kontext mit Ausreißern auf. In neuronalen Netzen zum Beispiel wird Overfitting häufig gezielt mit sogenannten 'Dropout'-Schichten bekämpft (siehe Sektion 2.3.7).

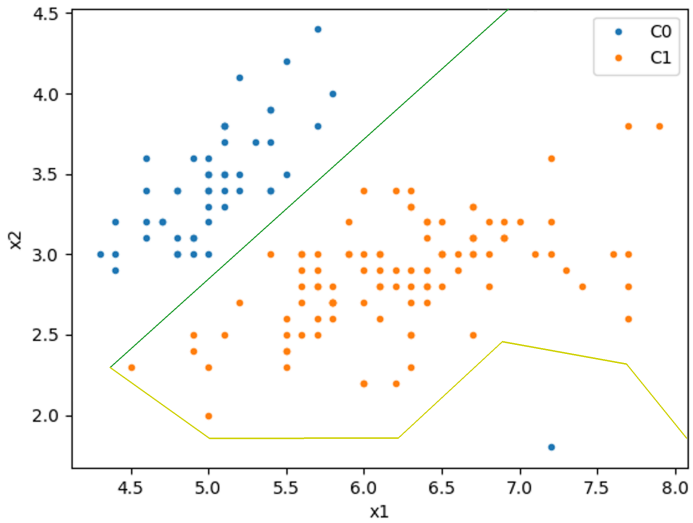


Abbildung 3: Beispiel für einen Ausreißer und Overfitting in einer Klassifikationsaufgabe

2.3.4 Features

Als 'Features' werden Merkmale bezeichnet, die man einem Lerner für jeden Datenpunkt übergibt - im Falle einer Person könnten das zum Beispiel Attribute wie Alter, Körpergröße oder ähnliches umfassen. Diese können in ihrer einfachsten Form die Rohdaten dieses Datenpunktes sein, für ein Bild zum Beispiel den RGB-Wert [6] jedes einzelnen Pixels eines Bildes. Es können aber beliebige und kompliziertere Features herangezogen werden, zum Beispiel die Positionierung des Bildes bei einer Google Suchanfrage.

Feature Engineering und Feature Learning

Den Prozess des Feature Engineerings beinhaltet die Auswahl und Erstellung von Features für ein ML-System durch Experten und Entwickler. Dies geschieht meist unter Zunahme von Fachwissen aus dem jeweiligen Anwendungsfeld. Während viele Methoden des maschinellen Lernens auf eine gute Auswahl von Features und somit ein gutes Feature Engineering angewiesen sind (so zum Beispiel die in dieser Arbeit behandelten Linear-Chain Conditional Random Fields), existieren mittlerweile auch viele automatische Methoden, die Features selbst aus den Rohdaten bestimmen. Auch neuronale Netzwerke sind in der Regel in der Lage, Features selbst zu bestimmten (vergleiche Sektion 2.3.7). Dies nennt sich Feature Learning [49].

2.3.5 Überwachtes / Unüberwachtes Lernen

ML-Systeme werden in 2 Kategorien unterschieden, die sich auf ihre Art, wie sie aus Daten lernen, beziehen: 'Überwacht' beziehungsweise 'Unüberwacht' (Supervised / Unsupervised). Während erstere für jede Einheit von Trainingsdaten ein Label benötigt und vor allem aus diesem Zusammenhang (Eingabedaten ⟷ Label) lernt, benötigen unüberwachte Lerner keine Labels - sie lernen hauptsächlich aus den Zusammenhängen, die sich rein aus den Eingabedaten erlernen lassen. So könnte es in dem Beispiel von oben möglich sein, dass ein Lerner rein aus der Art, wie die Hunde beziehungsweise Katzenbilder aufgebaut

[6] Als RGB-Wert bezeichnet man die Menge von Rot, Grün sowie Blau in einer Farbe

sind, Rückschlüsse darauf ziehen kann, welche sich ähnlich sind um sie so entsprechend zu gruppieren. Beispiele für überwachte Lerner sind Neuronale Netze sowie Linear-Chain Conditional Random Fields, welche auch im Rahmen der Arbeit betrachtet werden - dementsprechend werden nicht nur für das Testen, sondern auch für das Training der Modelle in dieser Arbeit gelabelte Daten benötigt. Für unüberwachte Lerner hingegen ist Clustering ein Beispiel.

2.3.6 Verlustfunktionen

Um aus Trainingsdaten zu lernen, benötigt ein überwachtes ML-System ein Maß, wie gut ein Modell ist - nur so ist es in der Lage, bessere von schlechteren Modellen zu unterscheiden. Denn ML-Systeme versuchen meistens, ein Optimierungsproblem zu lösen, in welchem sie die Verlustfunktion zu minimieren versuchen. Diese misst die Abweichung zwischen den vorgegeben Labels und den Vorhersagen, die das Modell für die jeweiligen Daten trifft. Der absolute Wert einer Verlustfunktion ist dabei, wie in allen Optimierungsproblemen, irrelevant - einzig das Verhältnis der Distanzen untereinander ist wichtig. Dementsprechend wären zum Beispiel $f(x) = x + 1$ sowie $g(x) = 2x + 2$ als Verlustfunktion äquivalent, da $g(x) = 2 \cdot f(x)$ gilt [75].
Allgemein existieren eine Vielzahl etablierter Verlustfunktionen, aus denen man auswählen kann. Abhängig von dem Feld der Anwendungen kann es aber auch sinnvoll sein, eine speziell auf das Problem zugeschnittene Verlustfunktion zu verwenden. Für den folgenden Abschnitt wird ein Modell als folgende Funktion betrachtet, welches für eine n-dimensionale Eingabe x eine m-dimensionale Vorhersage $h(x)$ berechnet:

$$\begin{aligned} n, m &\in \mathbb{N} \\ h &: \mathbb{R}^n \mapsto \mathscr{T}^m \end{aligned} \tag{2}$$

Betrachtet wird hier der, für die Arbeit relevante, Fall für die allgemeine Klassifikation $\mathscr{T} = [0, 1]$ - hierbei gilt $m = 1$ für die binäre Klassifikation, $m >= 3$ für eine Multiklassen-Klassifikation.
Viele Verlustfunktionen benutzen Vektornormen der Form $\|\cdot\| : \mathscr{T}^m \mapsto \mathbb{R}_0^+$, um die Abweichungen der Vektoren voneinander in skalare Werte zu transformieren - welche Norm genau verwendet wird, steht meist frei. Regelmäßig wird zum Beispiel die euklidische Norm (auch 2-Norm genannt) verwendet:

$$\begin{aligned} z &= (z_0, ..., z_{m-1}) \in \mathscr{T}^m \\ \|y\|_2 &= \sqrt{\sum_{i=0}^{m-1} z_i^2} \end{aligned} \tag{3}$$

Es ist dabei irrelevant, ob es sich bei den Ausgaben von h um diskrete Werte (zum Beispiel ausschließlich 0 oder 1) oder um kontinuierliche Werte (wie zum Beispiel bei einem Ranker (näheres dazu in Sektion 2.3.10)) handelt.
Die intuitivste Verlustfunktion ist der Mean Absolute Error (MAE, auch L1 genannt), welcher den Durchschnitt aller Abweichungen für die Inputs X von den gewünschten Ergebnissen Y darstellt. Eng mit ihr verwandt ist der Mean Squared Error (MSE, auch L2 genannt), der die Abweichungen quadriert - als Folge daraus werden größere Abweichungen bedeutend stärker gewichtet als kleinere (siehe Abbildung 4) - damit ist MSE anfälliger für Outlier (siehe 2.3.3) [41]. Den Multiklassenfall verarbeiten MAE sowie MSE mithilfe der Vektornorm.

$$\begin{aligned} X &= (x_0, ..., x_{n-1}) \\ Y &= (y_0, ..., y_{m-1}) \\ MAE(X, Y) &= \frac{1}{2 \cdot n} \sum_{i=0}^{n-1} \|(y_i - h(x_i))\| \\ MSE(X, Y) &= \frac{1}{2 \cdot n} \sum_{i=0}^{n-1} \|(y_i - h(x_i))\|^2 \end{aligned} \tag{4}$$

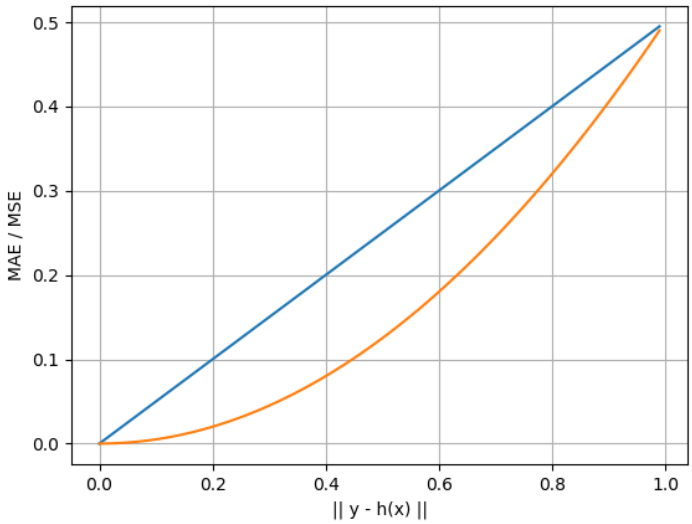


Abbildung 4: Werte von MAE und MSE abhängig vom Norm-Term im Vergleich

Eine im Bereich der Klassifikation mit am häufigsten genutzte Funktion, ist der Cross-Entropy Loss (CEL). Er berechnet sich für den allgemeinen Fall für m Klassen wie folgt [86]:

$$\begin{aligned} X &= (x_0, ..., x_{n-1}) \\ Y &= (y_0, ..., y_{m-1}) \\ CEL(X,Y) &= -\frac{1}{n}\sum_{i=0}^{n-1}\sum_{c=0}^{m-1} y_{i,c}\log(h(x_{i,c})) \end{aligned} \tag{5}$$

Für den Fall einer binären Klassifikation, welche die Klassen über einem einzelnen Output ($m = 1$) unterscheidet wie ein Ranker, wird häufig der Binäre Cross-Entropy Loss (BCEL) verwendet [86]:

$$BCEL(X,Y) = -\frac{1}{n}\sum_{i=0}^{n-1}[y_i \ln h(x_i) + (1-y_i)\ln(1-h(x_i))] \tag{6}$$

Allgemein können diese Verlustfunktionen in der Form $L = \frac{1}{n}\sum_{i=0}^{n-1} C(x_i, y_i)$ notiert werden, wobei die Funktion C die konkrete Verlustfunktion darstellt. Für die oberen Funktionen ergibt sich damit:

$$\begin{aligned} C_{MAE}(x,y) &= \frac{1}{2}\|(y - h(x))\| \\ C_{MSE}(x,y) &= \frac{1}{2}\|(y - h(x))\|^2 \\ C_{CEL}(x,y) &= -\sum_{c=0}^{m-1} y_c \log(h(x_c). \end{aligned} \tag{7}$$

Kosten-Sensitives Lernen

Die oben betrachteten Kostenfunktionen betrachten jede Art der Fehlklassifikation mit dem selben Gewicht. Dies ist aber nicht immer optimal: Als Beispiel nehme man die Diagnose einer Krankheit. Dort ist eine Fehlklassifikation, welche einen gesunde Patienten als 'krank' diagnostiziert, nur 'begrenzt schlimm': In weitergehenden Untersuchungen wird man feststellen, dass es sich um einen Fehler gehandelt hat und der Patient wird entlassen. Der umgekehrte Fall hingegen ist deutlich kritischer: Wird ein kranker Patient fälschlicherweise als 'gesund' eingestuft, werden keine weiteren Untersuchungen vorgenommen und

die Krankheit hat bis zu einer nächsten Untersuchung Zeit, sich auszubreiten. Dies muss ein ML-System berücksichtigen können, ohne alle Patienten einfach als 'krank' einzustufen - denn dann wäre nichts gewonnen. Um dies Umzusetzen, wird Kosten-Sensitives Lernen eingesetzt. Dafür gibt es mehrere Methoden:

Gewichtete Instanzen Wenn das ML-System dies unterstützt, können Beispiele entsprechend ihrer Kosten gewichtet gezählt werden - für die Klassifikation von Krankheiten würde man den 'krank'-Instanzen also ein höheres Gewicht geben.

Oversampling Beispiele, die höhere Kosten besitzen, werden anhand derer in entsprechender Anzahl kopiert - für die Diagnose von Krankheiten würde man also zum Beispiel alle 'krank'-Beispiele verdreifachen - dies ändert zum Beispiel auch die Steigung der Gerade bei der Optimierung im ROC-Space.

Gewichte in der Verlustfunktion In der Verlustfunktion eines ML-Systems werden Fälle, die höhere Kosten besitzen (zum Beispiel 'krank' als 'nicht krank' klassifiziert) stärker gewichtet als andere Fälle.

Oversampling kann auch eingesetzt werden, um unausgewogene Datensätze (welche zum Beispiel deutlich mehr positive als negative Instanzen enthalten) auszugleichen. Entwicklungs, -sowie Testset sollten aber immer repräsentativ für die originale Verteilung bleiben [85].

2.3.7 Neuronale Netze

(Künstliche) neuronale Netze (NN) haben in den letzten Jahren eine immer weiter steigende Popularität genossen - besonders auch in den Bereichen der Anonymisierung und Pseudonymisierung von personenbezogenen Daten sowie NER, in welchen hauptsächlich Rekurrente Neuronale Netze (RNNs) eingesetzt werden (mehr dazu in Sektion 3). Sie gehören zu den überwachten Lernmethoden.
Ein neuronales Netz besteht aus 2 grundlegenden Bauteilen: Neuronen (Nodes) sowie Verbindungen (Connections) zwischen ihnen. Mit ihrer Hilfe lassen sich Berechnungen durchführen: Jedes Neuron kann einen Wert annehmen, welcher durch die Werte der mit ihm verbundene Neuronen beeinflusst wird.
Ein neuronales Netz ist in Schichten aufgebaut: Jede Schicht (Layer) besteht aus einer grundsätzlich frei wählbaren Anzahl an Neuronen mit zur nächsten Schicht gerichteten Verbindungen - welche Neuronen mit welchen Verbunden werden ist grundsätzlich auch frei wählbar, in einem Basis-Aufbau werden alle Neuronen mit allen verbunden (vollständig verbunden) - anders ist dies zum Beispiel bei Convolutional Neural Networks (CNNs), wo nicht jedes Neuron mit jedem verbunden ist (mehr dazu im weiteren Verlauf dieser Sektion). Es existieren 3 Arten von Schichten: Die Eingangsschicht, welche die Werte von einem Datenpunkt übernimmt (das heißt diese Neuronen haben die Werte des Datenpunktes), die Ausgangsschicht, dessen Werte die finale Vorhersage darstellen, sowie sogenannte versteckten Schichten, welche zwischen Ein-und Ausgangsschicht platziert sind. Während jedes neuronale Netz eine Ein-sowie Ausgangsschicht hat, können beliebig viele versteckte Schichten, oder auch gar keine, verwendet werden - bei mehr als einer versteckten Schicht spricht man von einem tiefen neuronalen Netz (mehr dazu in Sektion 2.3.9). Der beispielhafte Aufbau eines neuronalen Netzes ist in Abbildung 5 zu sehen. Es berechnet die Vorhersage $y(x) = (y_1(x))$ für die Eingabe $x = (x_1, x_2)$.

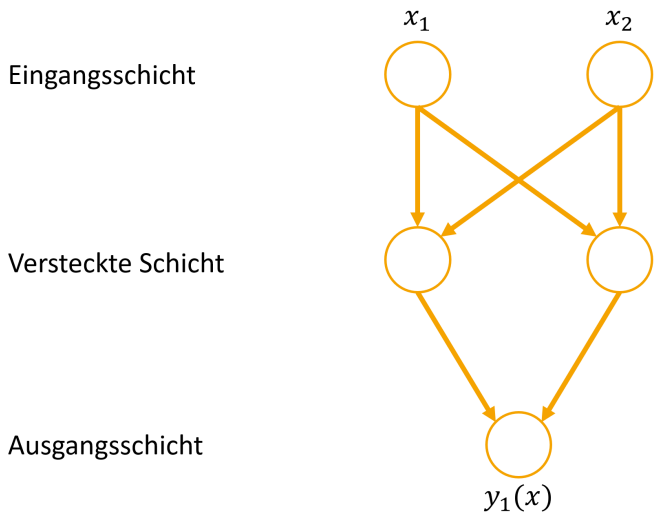


Abbildung 5: Der exemplarische Aufbau eines einfachen neuronalen Netzes

Allerdings sind noch zwei weitere Komponenten notwendig, um ein neuronales Netz richtig für Berechnungen verwenden zu können: Realwertige Gewichte, welche bestimmen, wie viel der Wert eines Neurons den Wert der mit ihm verbundenen Neuronen beeinflusst, sowie sogenannte Aktivierungsfunktionen, die bestimmen wie genau der Wert eines Neurons berechnet wird (mehr dazu unter Aktivierungsfunktionen). Eine weitere typische Komponente ist die Verwendung von Bias-Neuronen. Für jede Schicht, in welcher man ein Bias verwenden möchte, platziert man in der vorhergehenden Schicht ein Neuron, welches konstant den Wert 1 besitzt und keine eingehenden Verbindungen hat, dafür aber ausgehende Verbindungen zu allen Neuronen der nächsten Schicht besitzt. So ist es zum Beispiel für Neuronen, welche nur '0'-en als Eingangswert erhalten trotzdem möglich, Werte ungleich 0 anzunehmen [9].

Berechnungen mit einem neuronalen Netz

Wie genau all dies nun zur Berechnung von Ausgangswerten eingesetzt wird, zeigt folgendes Beispiel: In Abbildung 6 ist ein neuronales Netz dargestellt, welches sowohl für die versteckte,- als auch die Ausgangsschicht Bias-Neuronen (x_0 sowie z_0) benutzt. Als Aktivierungsfunktionen nutzt es Sigmoid ($sig(x) = \frac{1}{1+e^{-x}}$) und eine lineare Funktion $I(x) = x$, welche elementeweise auf einen Vektor angewendet werden - näheres dazu im Abschnitt Aktivierungsfunktionen. Jede Verbindung besitzt ein Gewicht der Form $w^{\{l\}}_{i_0 i_1}$, wobei l die Zielebene der Verbindung notiert, i_0 den Index des Zielneurons und i_1 den Index des Startneurons. In der Abbildung sind exemplarisch einige Gewichte angetragen. Für jede Ebene werden die Gewichte in einer Matrix $W^{\{l\}}$ gesammelt - für das Beispiel stellt sich dies wie folgt dar:

$$W^{\{Z\}} = \begin{pmatrix} w^{\{Z\}}_{10} & w^{\{Z\}}_{11} & w^{\{z\}}_{12} \\ w^{\{Z\}}_{20} & w^{\{Z\}}_{21} & w^{\{Z\}}_{22} \\ w^{\{Z\}}_{30} & w^{\{Z\}}_{31} & w^{\{Z\}}_{32} \\ w^{\{Z\}}_{40} & w^{\{Z\}}_{41} & w^{\{Z\}}_{42} \end{pmatrix} W^{\{Y\}} = \begin{pmatrix} w^{\{Y\}}_{10} & w^{\{Y\}}_{11} & w^{\{Y\}}_{12} & w^{\{Y\}}_{13} & w^{\{Y\}}_{14} \\ w^{\{Y\}}_{20} & w^{\{Y\}}_{21} & w^{\{Y\}}_{22} & w^{\{Y\}}_{23} & w^{\{Y\}}_{24} \end{pmatrix} \tag{8}$$

Gewichte der Form $w^{\{l\}}_{0i_1}$ tauchen nicht in der Matrix auf, da sie die Bias-Neuronen (Index 0) zum Ziel hätten - diese können als konstant 1 angesehen werden. Gewichte der Form $w^{\{l\}}_{i_0 0}$ haben Bias-Neuronen als Startpunkt und nennt man daher Bias-Gewichte oder Bias-Werte.

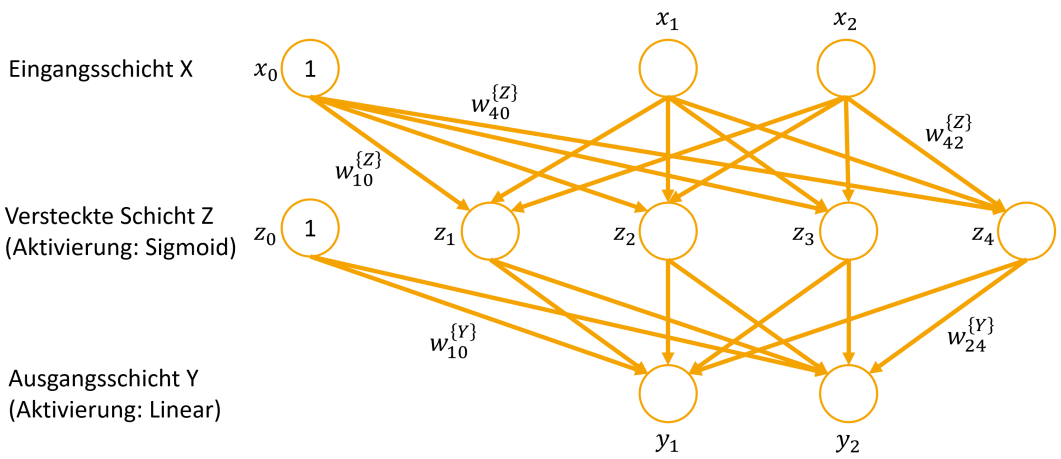


Abbildung 6: Beispiel für ein neuronales Netzwerk mit Bias-Neuronen

Die Werte der Neuronen werden Schichtenweise berechnet, beginnend bei der Eingangsschicht, welche schlicht die Werte der Eingangsvariablen übernimmt. Für jede weitere Schicht T ($n \in \mathbb{N}$ Neuronen, Aktivierungsfunktion λ_T), welche auf Schicht U mit $m \in \mathbb{N}$ Neuronen folgt werden die Werte folgendermaßen berechnet [9]:

$$
\begin{aligned}
&t_0 = u_0 = 1 \\
&\hat{T} = (t_1, ..., t_n)^T \\
&T = \begin{cases} \begin{pmatrix} t_0 \\ \hat{T} \end{pmatrix} & \text{falls die folgende Schicht einen Bias nutzt} \\ \hat{T} & \text{sonst} \end{cases} \\
&\hat{U} = (u_1, ..., u_n)^T \\
&U = \begin{cases} \begin{pmatrix} u_0 \\ \hat{U} \end{pmatrix} & \text{falls T einen Bias nutzt} \\ \hat{U} & \text{sonst} \end{cases} \\
&\hat{T} = f_{\hat{T},U} = \lambda_T(W^{\{T\}} \cdot U)
\end{aligned}
\tag{9}
$$

Dies lässt sich kombinieren zu folgender, allgemeinen Formel $f_{T,U}(W^{\{T\}})$:

$$
T = f_{T,U}(W^{\{T\}}) = \begin{cases} \begin{pmatrix} 1 \\ f_{\hat{T},U} \end{pmatrix} & \text{falls die folgende Schicht einen Bias nutzt} \\ f_{\hat{T},U} & \text{sonst} \end{cases}
\tag{10}
$$

Hat man nun Werte für die Gewichte gegeben (mehr Informationen dazu, wie diese Gewichte berechnet werden im Abschnitt 'Erlernen der Gewichte'), kann man mit dem neuronalen Netz Berechnungen durchführen - dies wird nun an dem Beispielnetzwerk aus Abbildung 6 erläutert. Dafür nehmen wir folgende, auf 2 Nachkommastellen gerundeten Gewichte, durch welche das neuronale Netz zu einem binären Addierer für 2 Stellen wird:

$$
W^{\{Z\}} = \begin{pmatrix} -1.98 & 5.44 & 5.44 \\ -1.98 & 5.44 & 5.43 \\ -3.17 & 2.11 & 2.11 \\ -2.99 & 1.96 & 1.97 \end{pmatrix} W^{\{Y\}} = \begin{pmatrix} -0.02 & -0.32 & -0.23 & 1.19 & 0.95 \\ -0.11 & 0.90 & 0.85 & -1.28 & -0.93 \end{pmatrix}
\tag{11}
$$

Nun kann man zum Beispiel anhand der obigen Definition (9) die Binärzahlen 1 sowie 0 addieren - dafür berechnet man zuerst die Werte für die Ebene Z und damit dann die Ausgangswerte (Ebene Y). (Aus visuellen Gründen werden alle dargestellten Werte auf 2 Nachkommastellen gerundet, es wird aber mit exakten Werten gerechnet)

$$\hat{X} = (1,0)^T \quad X = (1,1,0)^T$$

$$\hat{Z} = (z_1, z_2, z_3, z_4)^T \quad Z = \begin{pmatrix} z_0 \\ \hat{Z} \end{pmatrix}$$

$$\hat{Y} = Y = (y_1, y_2)^T$$

$$\hat{Z} = sig(W^{\{Z\}} \cdot X) = sig(\begin{pmatrix} -1.98 & 5.44 & 5.44 \\ -1.98 & 5.44 & 5.43 \\ -3.17 & 2.11 & 2.11 \\ -2.99 & 1.96 & 1.97 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}) = sig(\begin{pmatrix} -1.98 \\ -1.98 \\ -3.17 \\ -2.94 \end{pmatrix}) = \begin{pmatrix} 0.12 \\ 0.12 \\ 0.04 \\ 0.05 \end{pmatrix}$$

$$Z = \begin{pmatrix} 1 \\ 0.12 \\ 0.12 \\ 0.04 \\ 0.05 \end{pmatrix} \tag{12}$$

$$\hat{Y} = sig(W^{\{Y\}} \cdot Z) = I(\begin{pmatrix} -0.02 & -0.32 & -0.23 & 1.19 & 0.95 \\ -0.11 & 0.90 & 0.85 & -1.28 & -0.93 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0.12 \\ 0.12 \\ 0.04 \\ 0.05 \end{pmatrix}) = \begin{pmatrix} 0.00 \\ 1.00 \end{pmatrix}$$

Mit $Y = (0,1)^T$ erhalten wir das richtige Ergebnis der Addition der Bits 1 und 0 - dies funktioniert analog auch für alle anderen Eingaben ($(0,0)^T$, $(0,1)^T$, $(1,1)^T$).

Aktivierungsfunktionen

Eine Aktivierungsfunktion λ_l der Schicht l bestimmt, welchen Wert Neuronen in dieser Schicht für einen bestimmten Input annehmen (siehe Gleichung 9). Sie nehmen daher eine wichtige Position in neuronalen Netzen ein und haben einigen Einfluss darauf, ob und wie gut ein neuronales Netz lernt. Zu den häufig genutzten Aktivierungsfunktionen gehören die folgenden:

Linear $I_a(x) = a \cdot x Of\, ta = 1$

Sigmoid $sig(x) = \frac{1}{1+e^{-x}}$

ReLU $R(x) = max(0, x)$

Tanh $tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

Die Berechnung erfolgt dabei in der Regel für jedes Neuron isoliert von seinen Nachbarn der selben Schicht - daher ist die Berechnung dieser Funktionen für einen Vektor der Länge $n \in \mathbb{N}$ elementeweise definiert:

$$\lambda_l(\begin{pmatrix} x_0 \\ ... \\ x_{n-1} \end{pmatrix}) = \begin{pmatrix} \lambda_l(x_0) \\ ... \\ \lambda_l(x_{n-1}) \end{pmatrix} \tag{13}$$

Eine Ausnahme davon bildet die sogenannte 'Softmax' Aktivierung - für das Neuron j einer Schicht mit $k \in \mathbb{N}$ Neuronen wird sie wie folgt berechnet:

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{i=0}^{k-1} e^{x_i}} \tag{14}$$

Die 'Softmax' Aktivierung wird häufig in der Ausgangsschicht eingesetzt. Durch sie lassen sich die Ausgaben, zum Beispiel für den Fall einer Multiklassen-Klassifikation, in Wahrscheinlichkeiten für die einzelnen Klassen (zwischen 0 und 1) umwandeln, welche sich auf 1 aufaddieren. Eine Ausgabe wie [0.75, 0.2, 0.05] lässt sich im Rahmen einer Softmax-Aktivierung so interpretieren, dass das neuronale Netz zu 75% sicher ist, dass die eingegebenen Daten zu der ersten Klasse gehören.
Aktivierungsfunktionen werden darin unterschieden, ob sie lineare Funktionen darstellen (die Erste) oder nichtlineare (die anderen 4) - welche man einsetzt hat einen großen Effekt darauf, was das neuronale Netz lernen kann. Nutzt ein neuronales Netz ausschließlich lineare Aktivierungsfunktionen, kann es nur lineare Zusammenhänge erkennen und einfache Funktionen, wie zum Beispiel ein 'Exklusives Oder' (XOR) [7] nicht lernen - eine deutliche Einschränkung. Daher werden in der Regel nichtlineare Aktivierungsfunktionen eingesetzt [9].

Neuronale Netze als Klassifizierer

Um NNs in der Klassifikation einzusetzen, sind einige Punkte zu beachten, die nun für den Fall der binären-sowie der Multiklassen-Klassifikation erläutert werden.

Binäre-Klassifikation Die Klassen der Trainingsbeispiele werden als '0' sowie als '1' kodiert und dem NN zum Training übergeben - dieses sollte in der Ausgangsschicht (mit exakt einem Neuron) eine Aktivierung wie Sigmoid nutzen, welche nur Werte zwischen 0 und 1 ausgibt. Um Vorhersagen zu erhalten, übergibt man dem NN die entsprechenden Eingabedaten - dies wird nun einen Wert zwischen 0 sowie 1 ausgeben, wodurch das neuronale Netz einem Ranker entspricht, wie im Abschnitt 'Ranker im ROC-Space' (Sektion 2.3.10) erläutert.

Multiklassen-Klassifikation Die k Klassen der Trainingsbeispiele werden mithilfe einer One-Hot-Kodierung kodiert, sodass jedes Label einem Vektor der Länge k entspricht. Anschließend werden sie dem NN übergeben, welches in seiner Ausgangsschicht mit k Neuronen üblicherweise eine Softmax-Aktivierung verwendet. Um eine Vorhersage zu erhalten, übergibt man dem NN die entsprechenden Eingabedaten - die Klasse, welche dem Neuron mit dem höchsten Ausgabewert entspricht, entspricht der Vorhersage des NN.

Erlernen der Gewichte

Eine zu klärende Frage ist, wie man die Gewichte, die eben verwendet wurden, erhält - denn als Technik des Maschinellen Lernens ist es natürlich das Ziel, dass das neuronale Netz diese aus Beispielen lernen kann. Zu Beginn des Lernprozesses werden die Gewichte meist zufällig, mit Werten aus einem gewissen Intervall, initialisiert. Anschließend ist es das Ziel, die Verlustfunktion zu minimieren - dies geschieht durch die Verwendung eines Optimierers. Doch dieser benötigt den Gradienten der Kostenfunktion in Richtung jedes einzelnes Gewichtes - diese sind nicht trivial zu erhalten. Daher geschieht dies durch eine Kombination aus 2 Prozessen: Forward- sowie Backpropagation. Unter Forwardpropagation versteht man den Prozess, für bestimmte Eingangswerte die Ausgangswerte des neuronalen Netzes zu berechnen - wie in den Gleichungen 9 sowie 10 beschrieben und in Gleichung 12 exemplarisch durchgeführt. Anschließend wird der Fehler anhand der Verlustfunktion C (vergleiche 2.3.6) berechnet. Hierfür kann ein neuronales Netz mit $L \in \{l | \forall l \in \mathbb{N} : l \geq 2\}$ Schichten S_l als folgende Funktion betrachtet werden:

$$h_{NN}(W) = \prod_{l=0}^{L-2} f_{S_{l+1},S_l}(W^{\{l+1\}}) \tag{15}$$

Ist der Verlust berechnet, kommt der Prozess der Backpropagation ins Spiel: Zuerst wird der Fehler $\delta^{\{l\}}$ für jede Schicht l berechnet - jeder Eintrag in $\delta^{\{l\}}$ stellt den Fehler für das jeweilige Neuron dar. Die

[7] Eine XOR-Funktion ist nur dann gleich Eins, wenn die Menge der '1' in den Eingabewerten ungerade ist - andernfalls ist ihr Ergebnis Null

Berechnung geschieht zuerst für die letzte Schicht, danach wird er rückwärts (Backwards) durch alle Schichten hindurch bis zur Eingangsschicht hin berechnet [9]. Für die letzte Schicht $L-1$ berechnet sich der Fehler für die Kostenfunktion C wie folgt:

$$\begin{aligned} z^{\{l\}} &= W^{\{l\}} \cdot S_{l-1} \\ a^{\{l\}} &= \lambda_{L-1}(z^{\{l\}}) \\ \delta^{\{L-1\}} &= \nabla_{a^{\{L-1\}}} C(a^{\{L-1\}}, y) \odot \nabla_{z^{\{L-1\}}} \lambda_{L-1}(z^{\{L-1\}}) \end{aligned} \tag{16}$$

Hierbei notiert $a_i^{\{l\}}$ die Aktivierungen aller Neuronen der Schicht l, basierend auf den gewichteten, eingehenden Verbindungen $z^{\{l\}}$. Durch y wird das Label des jeweiligen Beispiels, welches in der Forwardpropagation berechnet wurde, angegeben. $\odot$ notiert das Hadamard Produkt, welches die Komponentenweise Multiplikation zweier gleich großer Vektoren beziehungsweise Matrizen darstellt. Die Kostenfunktion muss $C(x, y)$ muss differenzierbar sein - dies ist für die in Gleichung 7 vorgestellten Funktionen aber gegeben. Für die davor liegenden Schichten berechnet sich $\delta^{\{l\}}$ wie folgt:

$$\delta^{\{l\}} = ((W^{l+1})^T \cdot \delta^{\{l+1\}}) \odot \nabla_{z^{\{l\}}} \lambda_l(z^{\{l\}}) \tag{17}$$

Um dies nun zur Optimierung der Gewichte zu nutzen, benötigen wir noch für jedes Gewicht den Gradienten $\frac{\partial C}{\partial w_j^{\{l\}}{}_k}$. Dieser ist gegeben durch [9]:

$$\frac{\partial C}{\partial w_{jk}^{\{l\}}} = a_k^{\{l-1\}} \cdot \delta_j^{\{l\}} \tag{18}$$

Hyperparameter

Ein neuronales Netzwerk bietet neben der Anzahl der Schichten, der Art der Aktivierungsfunktionen sowie der Anzahl der Neuronen in jeder Schicht noch einige weitere wichtige Hyperparameter, die in diesem Abschnitt kurz vorgestellt werden.

Batch-Size In neuronalen Netzen werden Beispiele meistens in Gruppen, sogenannten Batches, verarbeitet - die gewählte Größe der Gruppen ist hierbei abhängig von der Anwendung, sie kann zum Beispiel 64, 256 oder 1024 betragen. Für alle Beispiele in einem Batch werden gemeinsam Forward-sowie Backpropagation (eine Iteration) ausgeführt. Während oben der Fall für ein Beispiel (Batch-Size = 1) erklärt wurde, lässt sich dies einfach auf mehrere Beispiele übertragen: Für die Forwardpropagation, zum Beispiel, nutzt man als Input eine Matrix, welche pro Spalte je einen Datenpunkt beinhaltet [22].

Epochs Die Anzahl der sogenannten Epochen wird durch den Epochs-Wert festgelegt. Jede Epoche stellt hierbei einen kompletten Durchlauf aller Trainingsbeispiele (für jedes Beispiel einmal Forward- sowie Backpropagation) dar [22].

Optimierer Es können verschiedene Optimierungsmethoden angewendet werden, um die Gewichte eines neuronalen Netzes zu optimieren. Häufig genutzte Methoden sind zum Beispiel Adam, Adagrad oder RMSProp - sie alle haben gemeinsam, dass sie Gradienten-Basiert arbeiten, da es gilt, das Ergebnis der Backpropagation (den Gradienten) zu nutzen. Jeder Optimierer besitzt wiederum eigene Parameter, wie zum Beispiel die sogenannte Lernrate, welche bestimmt wie stark Anpassungen an den Gewichten vorgenommen werden - ein zu kleiner Wert hat eine lange Lerndauer zur Folge (teilweise kann es als Folge dessen auch zu gar keinem Lernerfolg führen), ein zu großer Wert kann für Instabilitäten sorgen, da die Anpassungen an die Gewichte zu groß sind um das Optimum zu finden [22].

Dropout Die Technik des Dropout wird eingesetzt, um in bestimmten Schichten eine gewisse Anzahl an Neuronen künstlich während des Trainings zu deaktivieren. Diese Technik wurde 2014 von Srivastava et al. vorgestellt und hat sich seitdem als erfolgreich bewiesen, Overfitting zu verringern [77].

Während bei anderen ML-Methoden häufig Feature Engineering notwendig ist, können neuronale Netze nützliche Features größtenteils selbst aus Daten isolieren und weniger nützliche Features 'ausblenden', in dem die zugehörigen Neuronen wenig Gewicht erhalten. Features, welche sich nicht aus den Rohdaten ergeben, sollten hingegen dem neuronalen Netz übergeben werden, da es nicht in der Lage sein kann, diese aus den Rohdaten zu lernen [9].
Neuronale Netze lernen zu Beginn des Lernprozesses in der Regel sehr schnell, danach flacht die Lernkurve ab und Verbesserungen geschehen nur noch in kleineren Schritten. Ein Beispiel für solch eine Lernkurve kann in Abbildung 7 gefunden werden.

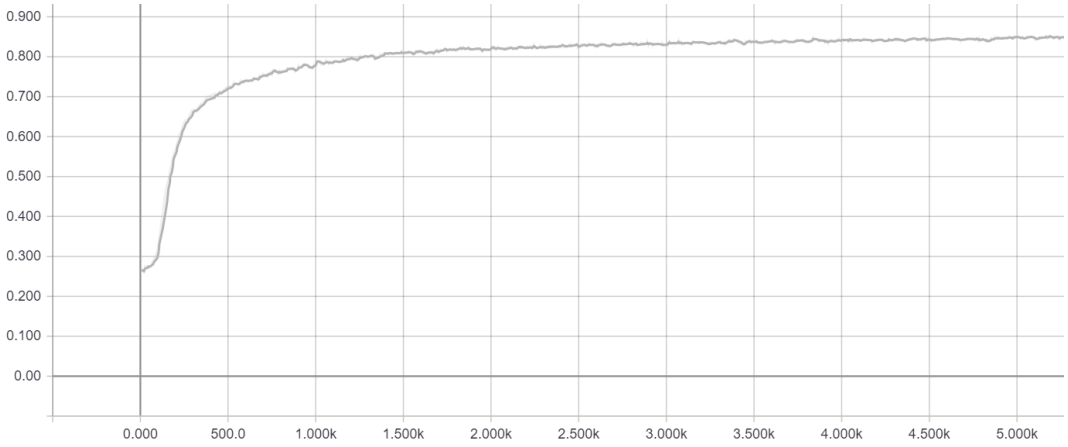


Abbildung 7: Beispiel für eine typische Lernkurve eines neuronalen Netzes: Die horizontale Achse notiert die Anzahl der Iterationen, die Vertikale die Accuracy zu einem gegeben Zeitpunkt

Rekurrente Neuronale Netze (RNNs)

Eine spezielle Art von neuronalen Netzen sind rekurrente neuronale Netze (RNNs). Im Gegensatz zu gewöhnlichen neuronalen Netzen, aufgebaut aus vollständig verbundenen Schichten, basiert deren Ergebnis nicht nur auf der momentanen Eingabe (Zeitpunkt t), sondern auch auf den vorhergehenden Daten (Zeitpunkte $< t$) - meist wird dies auf ganzen, sogenannte rekurrenten, Schichten eingesetzt und durch eine Verbindung zur Schicht selber symbolisiert, wie in Abbildung 8 dargestellt. Hier basiert der Ausgabewert von Schicht S_1^t nicht nur auf den eingehenden Werten $z^{\{1\}} = W^{\{1\}} \cdot S_0^t$, sondern auch auf

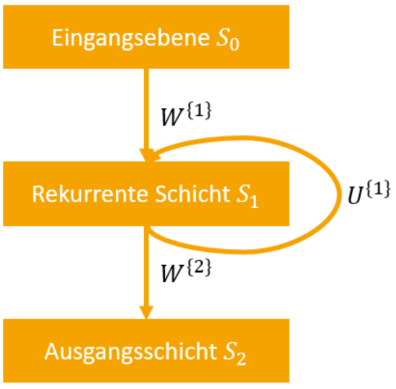


Abbildung 8: Exemplarischer Aufbau eines RNNs

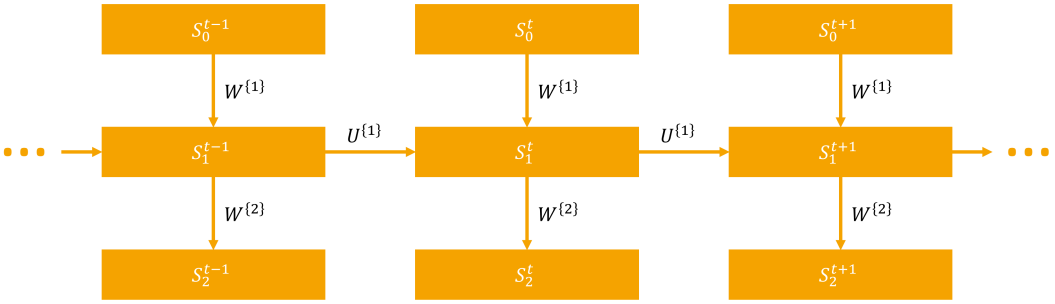


Abbildung 9: Darstellung eines entfalteten RNNs

den Werten der vorhergegangenen Werten S_1^{t-1}, gewichtet durch $U^{\{1,1\}}$ (S_l^t notiert hierbei den Wert der Schicht l zum Zeitpunkt t). Diese werden durch Addition kombiniert:

$$S_1^t = \lambda_1(W^{\{1\}} \cdot S_0^t + U^{\{1,1\}} \cdot S_1^{t-1}) \tag{19}$$

Die anderen, nicht rekurrenten Schichten werden wie gehabt berechnet.
Zur Veranschaulichung der Abhängigkeiten zwischen den Schichten lässt sich das Netzwerk auch grafisch auf-falten, wie Abbildung 9 zeigt. Dies veranschaulicht auch, dass der Wert von S_1^t direkt zwar nur zusätzlich von S_1^{t-1} abhängt, aber indirekt auch von den Werten S_1^{t-2}, S_1^{t-3}. Dies lässt sich auch mathematisch darstellen:

$$S_1^t = \lambda_1(W^{\{1\}} \cdot S_0^t + U^{\{1,1\}} \cdot \lambda_1(W^{\{1\}} \cdot S_0^{t-1} + U^{\{1,1\}} \cdot \lambda_1(W^{\{1\}} \cdot S_0^{t-2} + U^{\{1,1\}} \cdot S_1^{t-3}))) \tag{20}$$

Diese Art einer rekurrenter Schicht nennt man vollständig rekurrent, da alle Neuronen der zeitlich vorhergehenden Schicht mit allen des aktuellen Zeitpunktes verbunden sind.
Rekurrente Netzwerke (neuronale Netzwerke mit mindestens einer rekurrenten Verbindung) sind besonders gut geeignet für die Verarbeitung von Sequenzen, wie zum Beispiel natürliche Sprache (Sequenz von Wörtern und Satzzeichen). Eine Anwendung hierfür ist zum Beispiel, das neuronale Netz immer das nächste Wort vorhersagen zu lassen um so eine automatische Vervollständigung wie bei einer Google-Suche zu ermöglichen [70]. Dies ist in Abbildung 10 exemplarisch dargestellt. Es ist auch möglich, mehrere rekurrente Schichten nacheinander einzusetzen.
Neben vollständig rekurrenten Schichten existieren noch eine Vielzahl weiterer Arten von rekurrenten neuronalen Netzen - häufig werden dabei sogenannte Long-Short-Term-Memory (LSTM) sowie Gated-Reccurent-Units (GRUs) Zellen eingesetzt - jedes Neuron in der jeweiligen rekurrenten Schicht wird durch eine Kopie dieser Zellen ersetzt. Durch ihren inneren Aufbau ergeben sich besondere Eigenschaften.

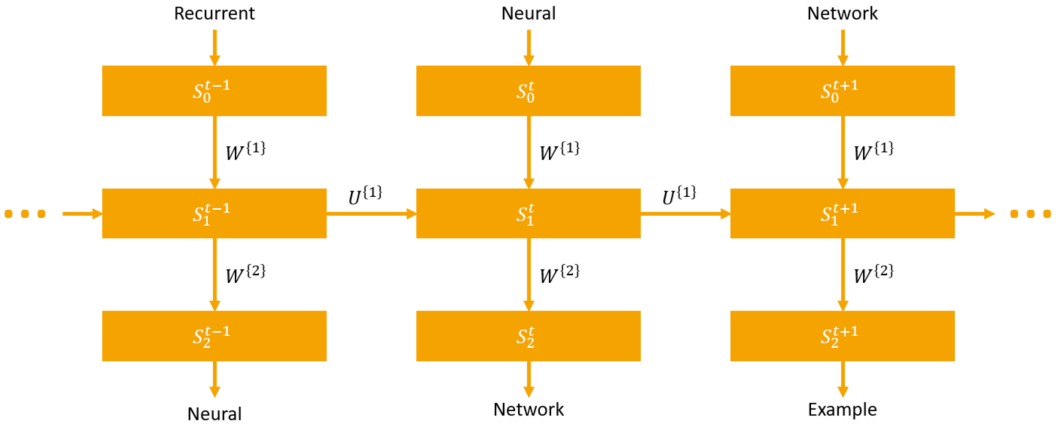


Abbildung 10: Beispiel für die Verarbeitung von natürlicher Sprache mit einem Rekurrentem neuronalen Netz

Long-Short-Term-Memory (LSTM): 1997 von Hochreiter et al. vorgestellt und 1999 von Gers et al. weiterentwickelt, werden LSTMs besonders in den letzten Jahren immer häufiger eingesetzt. Zu Deutsch übersetzt sich der Name zu "langes Kurzzeitgedächtnis", was dessen Aufbau geschuldet ist: Eine LSTM-Zelle besteht aus einem inneren Neuron (die sogenannte Zustandszelle), sowie aus 3 sogenannten Gates (ebenfalls Neuronen) [39] [31]:

- **Input-Gate** (auch Update-Gate genannt) bestimmt das Ausmaß, in dem ein neuer Input in den Wert des Zellzustandes einfließt
- **Reset-Gate** (auch Forget-Gate genannt) steuert das Ausmaß, in dem der Zellzustand in der Zelle verbleibt beziehungsweise vergessen wird
- **Output-Gate** bestimmt das Ausmaß, in dem der Zellzustand zu dem Ausgang, sprich dem Eingang der nachfolgenden Neuronen, beiträgt

Ein exemplarischer Aufbau ist in Abbildung 11 dargestellt. Zu beachten ist hierbei, dass verschiedene Ansätze existieren, wie sich die Werte der Neuronen untereinander genau beeinflussen - sie alle aber haben den selben, grundlegenden Aufbau gemeinsam. Häufig werden folgende Formeln verwendet: [54]

$$\begin{aligned}
\mathbf{i}_t &= sigmoid\left(\boldsymbol{W}_i\mathbf{h}_{t-1} + \boldsymbol{U}_i\mathbf{x}_t + \boldsymbol{b}_i\right) \\
\mathbf{f}_t &= sigmoid\left(\boldsymbol{W}_f\mathbf{h}_{t-1} + \boldsymbol{U}_f\mathbf{x}_t + \boldsymbol{b}_f\right) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\boldsymbol{W}_{c-1} + \boldsymbol{U}_c\mathbf{x}_t + \boldsymbol{b}_c) \\
\mathbf{o}_t &= sigmoid\left(\boldsymbol{W}_o\mathbf{h}_{t-1} + \boldsymbol{U}_o\mathbf{x}_t + \boldsymbol{b}_o\right) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right)
\end{aligned} \tag{21}$$

Hierbei bezeichnet t den aktuellen Zeitpunkt, b den jeweiligen Bias Wert des Neurons. Ein allgemeiner, exemplarischer Aufbau ist in Abbildung 11 zu sehen.

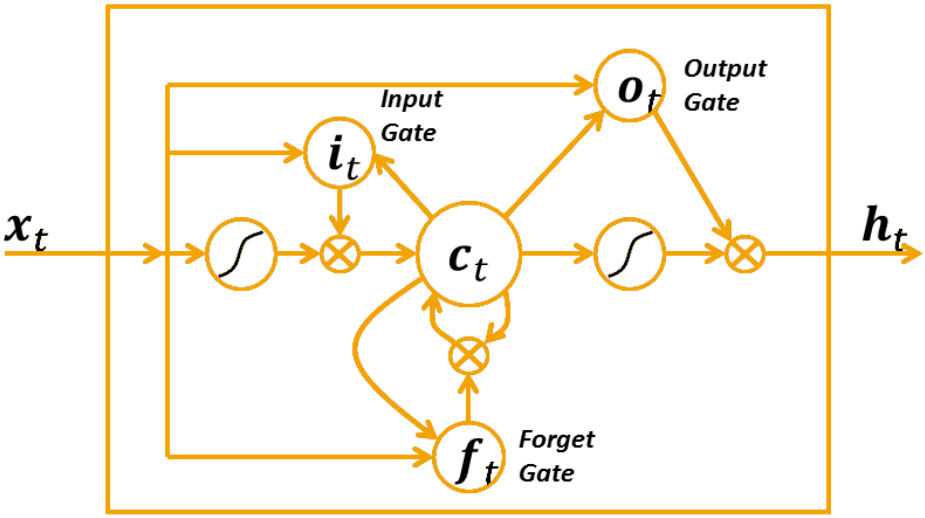


Abbildung 11: Exemplarischer Aufbau einer LSTM-Zelle, wobei c_t die Zustandszelle und h_t die Ausgabe zum Zeitpunkt t bezeichnet [8](Auf die Darstellung von Bias-Werten wurde verzichtet)

LSTM-Netzwerke finden mittlerweile eine sehr breites Anwendungsfeld - gerade auch in den Bereichen der Anonymisierung und NER, wie in Sektion 3 näher betrachtet wird.

[8] Bildquelle: `https://commons.wikimedia.org/wiki/File:Long_Short_Term_Memory.png`

Gated-Reccurent-Units (GRUs): Einen alternative Ansatz zu LSTMs stellt die 2014 von Cho et al. vorgestellten Gated-Reccurent-Units (GRUs) dar: Im Gegensatz zu LSTMs nutzen sie eine verringerte Anzahl an Gates (sie besitzen kein Output-Gate) und einen leicht veränderten Aufbau. Ähnlich zu LSTMs existieren verschiedene Ausprägungen des Basisaufbaus (Fully Gated), sowie eine noch kompaktere Version (Minimal Gated), in welcher sowohl das Input als auch das Reset-Gate in ein Forget-Gate zusammen gefasst werden [18]. Dies resultiert in einer im Vergleich zu LSTMs meist ähnlicher Leistung, wobei sie auf kleineren Datensätzen im Durchschnitt bessere Leistungen zeigen [19].
In den Bereichen der NER sowie der Anonymisierung werden jedoch fast ausschließlich LSTMs eingesetzt (vergleiche Sektion 3), weswegen GRUs in dieser Arbeit keine Anwendung finden.

Bidirektionale Rekurrente neuronale Netze (BRNNs): Betrachtet man Sequenzen mit einem, wie oben beschriebenen, unidirektionalen RNN, kann das neuronale Netz zur Bestimmung des Ergebnisses zum Zeitpunkt t nur auf die Werte $t' < t$ zurückgreifen, wie in Gleichung 20 dargestellt. Bidirektionale RNNs hingegen, wie 1997 von Schuster et al. vorgestellt, zeichnen sich dadurch aus, dass sie sowohl auf 'zukünftige', als auch auf 'vergangene' Eingangswerte zurückgreifen können. Somit stehen einem BRNN mehr Informationen pro Datenpunkt zur Verfügung, um Vorhersagen zu machen. Erreicht wird dies durch eine Kombination von sogenannten 'Vorwärtsschichten' (wie sie in normalen RNNs verwendet werden) sowie einer zusätzlichen 'Rückwärtsschicht'. Zu beachten ist hierbei, dass diese Schichten untereinander nicht direkt verbunden sind - daher werden BRNNs häufig auch als eine Kombination aus 2 RNN-Zellen betrachtet, wobei eine die Eingaben vorwärts, die andere die Eingaben rückwärts betrachtet. Daher ist es generell auch möglich, unterschiedliche Architekturen in den jeweiligen Schichten einzusetzen. So könnte beispielsweise die Vorwärtsschicht aus LSTM-Zellen bestehen, die Rückwärtsschicht aus GRU-Zellen. In der Regel aber wird eine einheitliche Art von Zellen für beide Schichten verwendet. Diese Netzwerke werden dann entsprechend auch als BLSTM beziehungsweise BGRU bezeichnet [74].
Der Aufbau ist in Abbildung 12 dargestellt, wobei S_1 die Forwärtsschicht und S_2 die Rückwärtschicht repräsentiert.

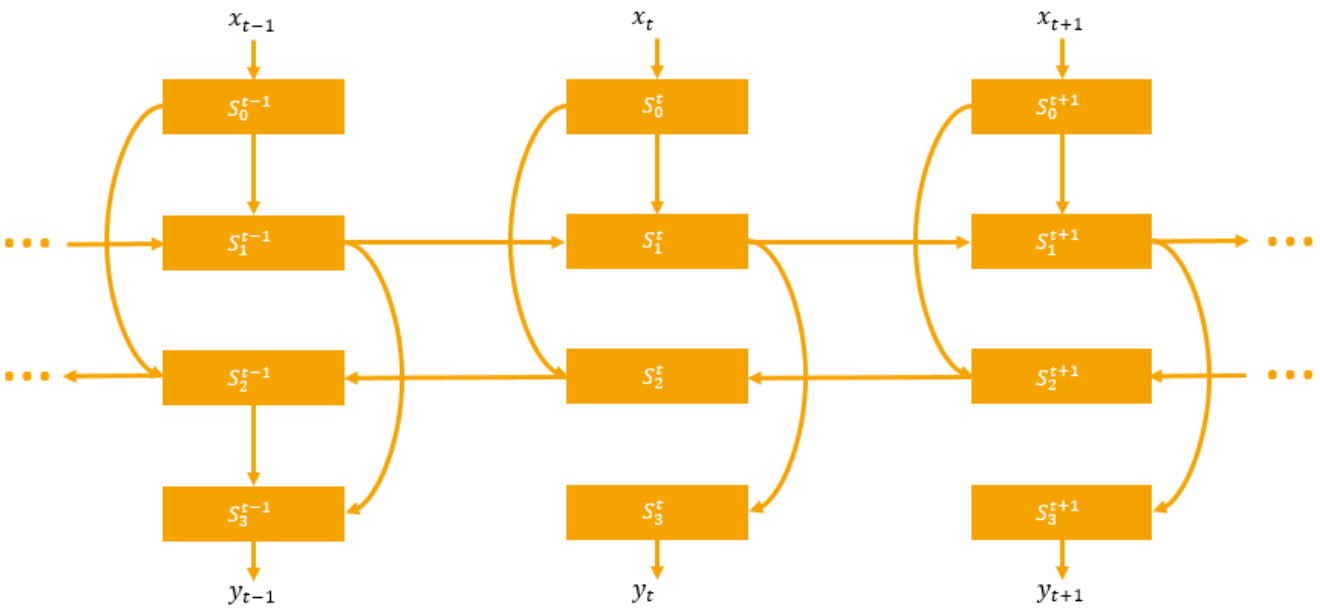


Abbildung 12: Exemplarischer Aufbau eines BRNNs

Convolutional Neural Networks (CNNs)

Eine weitere Art von neuronalen Netzen sind Convolutional Neural Networks (CNNs). Sie werden meist für die Verarbeitung von Bildern eingesetzt, doch in Kombination mit BRNNs werden sie auch im Rahmen von NER verwendet [17] (mehr dazu in Sektion 3.2.2). Über dieses Gebiet finden sie auch Einsatz in dieser Arbeit (vergleiche Sektion 4.3).

CNNs nutzen zwei verschiedene Arten von Schichten: Convolution-sowie Pooling-Schichten. Hierbei folgen auf eine, oder mehrere, Convolution-Schichten in der Regel eine Pooling-Schicht, welche eine Einheit bilden - vor der Pooling-Schicht können sich auch vollständig verbundene Schichten befinden. Solche Einheiten lassen sich prinzipiell beliebig häufig hintereinander einsetzen [33].

Convolution-Schichten: Um die Motivation hinter dem Einsatz von Convolution-Schichten zu verstehen, folgt zunächst ein Beispiel aus der Bildverarbeitung: Will man Hunde auf Bildern erkennen, kann man ein NN mit einem Satz von Bildern trainieren, auf welchem Hunde markiert sind. Typischerweise werden Bilder Pixel für Pixel an ein NN übergeben, sodass immer ein oder mehrere Neuronen der Eingangsschicht die Werte für genau einen Pixel erhalten. Das NN versucht nun, über alle Bilder hinweg Muster zu erkennen, welche einen Hund 'ausmachen' - doch auf den Bildern werden Hunde nicht immer an der selben Stelle auftauchen, sondern an verschiedenen: Mal oben, mal unten, mal rechts, mal links (vergleiche Abbildung 13). Da die Pixel alle an verschiedene Neuronen mit verschiedenen Gewichten übergeben werden, sind immer andere Verbindungen zwischen Neuronen zuständig, Hunde auf Bildern zu erkennen - obwohl sich das Muster, nach dem ein Hund erkannt werden kann, zwischen den Positionen in der Regel unverändert bleibt. Dabei besitzen vollständig verbundene Schichten oft große

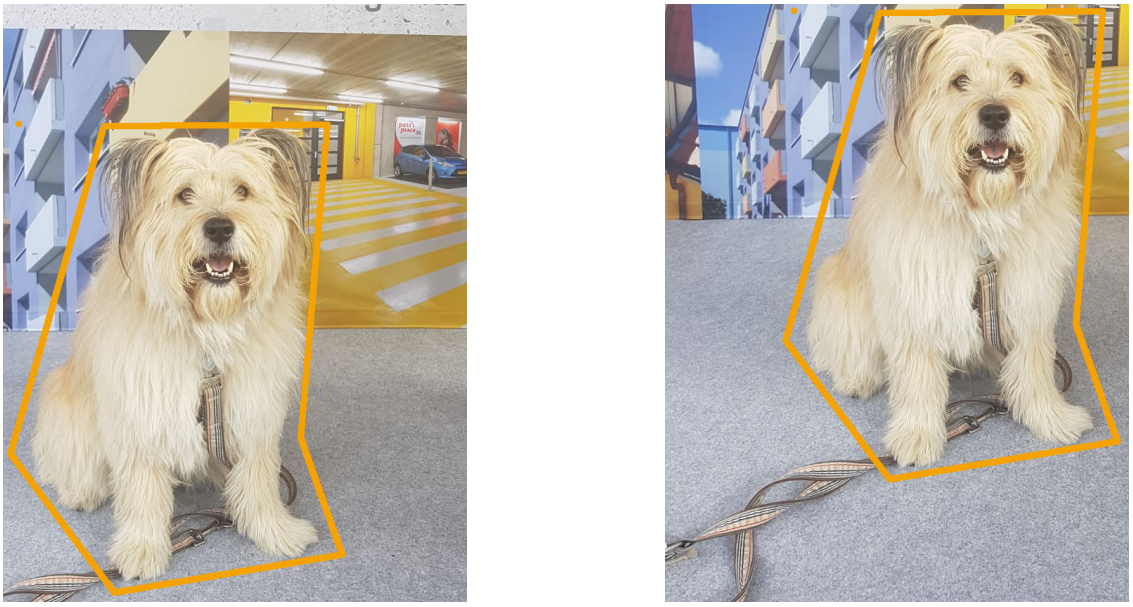


Abbildung 13: Zwei verschiedene Ausschnitte eines Hundebildes im Vergleich

Mengen an Gewichten, da jedes Neuron mit jedem verbunden ist. Selbst falls man für die Bilder eine vergleichsweise geringe Auflösung von $n = 64$ Schwarz-Weiß Pixeln je Seite wählt, benötigt die Eingangsschicht $m = n^2 = 4096$ Neuronen (jedes Neuron erhält den Schwarz-Weiß Wert eines einzelnen Pixels). Folgt darauf eine Schicht mit ebenso vielen Neuronen, resultiert dies bei vollständiger Verbundenheit in $m \cdot m = 16.777.216$, welche jeweils einzeln gespeichert und optimiert werden. Doch wie das obige Beispiel gezeigt hat, würden sich in manchen Anwendungsfällen, wie der Objektdetektion, viele Gewichte während des Trainings angleichen, um die selben Muster unabhängig ihrer Position erkennen zu können. Daher teilen sich Neuronen in Convolution-Schichten einen Kernel mit k' Gewichten, welche für alle Neuronen der Schicht zusammen trainiert werden - da so Muster unabhängig von ihrer Position zum Training der selben Gewichte beitragen sind CNNs in der Lage, für solche Aufgaben ein generelleres Modell zu entwickeln als ähnliche vollständig verbundene Netzwerke [33]. Des weiteren wird ein ausgehendes Neuron solch einer Schicht von maximal k' Neuronen der Eingabe 'beeinflusst', wodurch innerhalb dieser Schichten die Position von eingehenden Mustern erhalten bleibt.

Für eine Convolution Schicht T mit einem Kernel K sowie dessen Länge k', lässt sich die Berechnung wie in Definition 9 gegeben durchführen. Einzig die Definition der Gewichte $w_{ij}^{\{T\}}$ ändert sich wie folgt:

$$\begin{aligned} K &= (k_0, ..., k_{k'-1})^T \\ h &= \left\lfloor \frac{k'}{2} \right\rfloor \\ w_{ij}^{\{T\}} &= \begin{cases} k_{j-i+h} & \text{falls} j-i+h > 0 \wedge j-i+h < k' \\ 0 & \text{sonst} \end{cases} \end{aligned} \tag{22}$$

Zu beachten ist hierbei, dass in diesem Fall Padding-Werte von 0 verwendet werden. Für ein Padding mit alternativen Werten wird die Eingehende Schicht auf jeder Seite um h dieser Werte ergänzt und die Matrix $W^{\{T\}}$ entsprechend erweitert.
Der Aufbau einer Convolution-Schicht ist in Abbildung 14 exemplarisch für einen Kernel der Größe 3 dargestellt.

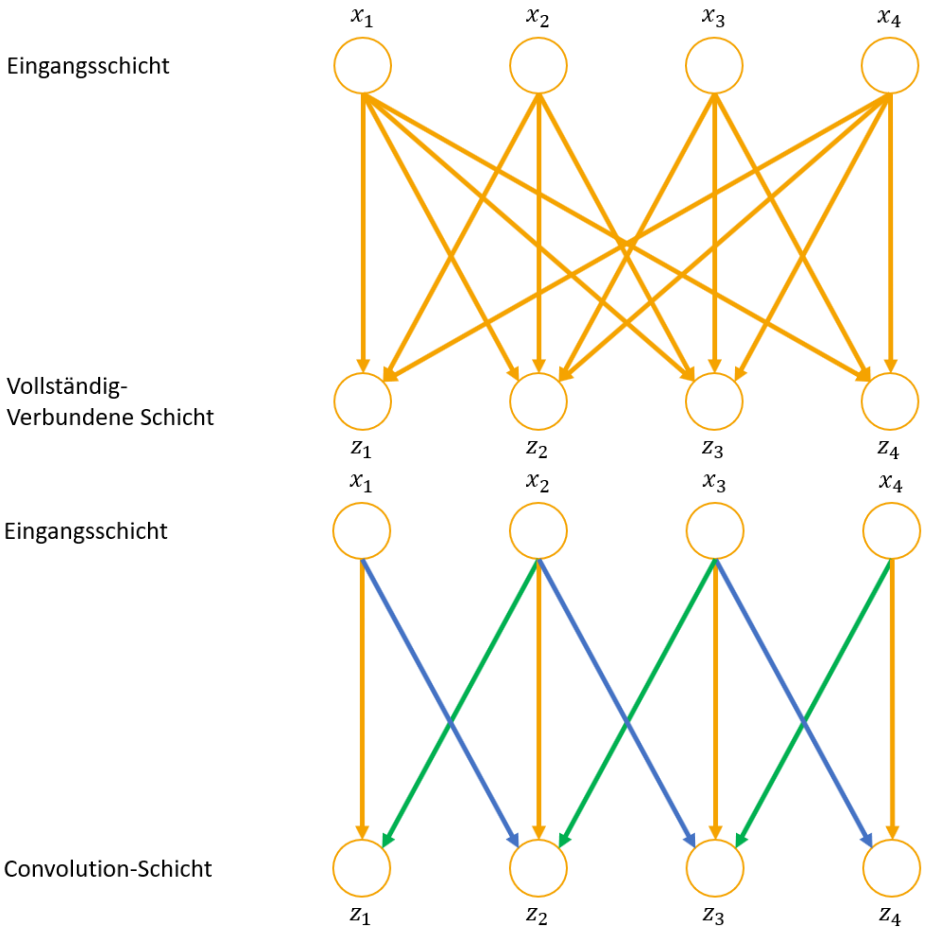


Abbildung 14: Vergleich einer Convolution-Schicht (Kernelgröße 3) mit einer Vollständig-Verbundenen Schicht. Im Falle des Convolution-Beispiels teilen sich Kanten mit der selben Farbe das Gewicht - fehlende Gewichte am Rand werden durch Padding-Methoden ergänzt

Pooling-Schichten: Pooling-Schichten haben die Aufgabe, die Ausgabe der Convolution-Schichten zu komprimieren um nur die wichtigsten Informationen zu erhalten. Dadurch ergibt sich in der Praxis die Möglichkeit, mit der gleichen Rechenleistung größere Netzwerke zu trainieren, welche aber komplexere Zusammenhänge erkennen können und so trotz des Informationsverlustes bessere Ergebnisse liefern. Außerdem hat sich gezeigt, dass sie dazu beitragen Overfitting zu verringern [33].

Im Rahmen des Poolings werden die Werte von einer gegebenen Menge an Neuronen mithilfe eines Pooling-Kernels zu einem einzelnen Wert zusammen gefasst. Es existieren hierbei verschiedene Formen, zu den meistgenutzten gehören Max,-sowie Mean-Pooling. Bei Max-Pooling ist das Ergebnis des Kernels der maximale Wert aller betrachteten Neuronen, beim Mean-Pooling deren Durchschnitt. In der Praxis wird meist Max-Pooling verwendet, denn es führt in der Regel zu besseren Ergebnissen [72].

2.3.8 Linear-Chain Conditional Random Fields

Conditional Random Fields (CRFs) sind ungerichtete, probabilistische graphische Modelle [9]. In dieser Sektion werden die sogenannten Linear-Chain CRFs (LCRFs) eingeführt, welche Sequenzen verarbeiten können und daher im NLP sowie in der Anonymisierung und Pseudonymisierung von personenbezogenen Daten als auch in der NER häufig eingesetzt werden (vergleiche Sektion 3). Sie sind eine besondere Form von CRFs. Für den restlichen Teil der Arbeit werden auch durch die Bezeichnung CRF(s) Linear-Chain CRF(s) referenziert.
LCRFs sagen für eine Sequenz $x \in S^n$ der Länge n die Sequenz $\hat{y} \in T^n$ als Label vor raus. Dabei besteht $\hat{y}$ selbst aus einer Sequenz an Labels aus T, welches jeweils das zugehörige (durch die Position bestimmte) Element aus S von x labelt. Dabei wird $\hat{y}$ so gewählt, dass es aus allen möglichen Sequenzen von Labels y die höchste bedingte Wahrscheinlichkeit, gegeben der Eingangssequenz x sowie Parametern λ, aufweist.

$$\hat{y} = argmax_y P(y|x, \lambda) \tag{23}$$

Zentraler Bestandteil der bedingten Wahrscheinlichkeit von LCRFs sind die sogenannten Feature-Funktionen (welche nur bedingt mit der Definition aus 2.3.4 übereinstimmen). Jede dieser Feature-Funktionen trifft Aussagen über das Verhältnis des Labels y_t gegenüber dem Wort an der momentanen Position, indiziert durch den Index t. Als Grundlage für diese Aussage erhält jede Feature Funktion 4 Parameter:

- Alle Labels $y \in T^n$
- Die gesamte Eingangssequenz $x \in S^n$
- Die Position $t \in \mathbb{N}$

Das Ergebnis jeder Feature-Funktion kann beliebig realwertig sein - oft beschränken sie sich aber auf $\{0, 1\}$. Ein LCRF besteht aus mehreren ($K \in \mathbb{N}$) Feature-Funktionen f_k, gesammelt in der Menge $\mathscr{F}$:

$$\mathscr{F} := \{f_k | \forall k \in \mathbb{N}, k < K\} \subseteq T^n \times S^n \times \mathbb{N} \mapsto \mathbb{R} \tag{24}$$

Ein weiterer, wichtiger Bestandteil sind die Gewichte λ_k, welche jeweils einer Feature-Funktion f_k zugeordnet sind - sie entscheiden, wie stark und in welcher Form eine Feature-Funktion die Bedingte Wahrscheinlichkeit beeinflusst (näheres dazu in den nächsten beiden Abschnitten) [80].

Bedingte Wahrscheinlichkeit $P(y|x, \lambda)$

Die Bedingte Wahrscheinlichkeit eines LCRF führt alle Komponenten zusammen und berechnet sich wie folgt:

$$\begin{aligned} \lambda &= (\lambda_0, ..., \lambda_{K-1}) \\ Z(x) &= \sum_{y \in Y} exp(\sum_{t=0}^{n-1} \sum_{k=0}^{K-1} \lambda_k \cdot f_k(y, x, t)) \\ P(y|x, \lambda) &= \frac{1}{Z(x)} \cdot exp(\sum_{t=0}^{n-1} \sum_{k=0}^{K-1} \lambda_k \cdot f_k(y, x, t)) \end{aligned} \tag{25}$$

[9] Probabilistische Graphische Modelle sind Graphen, deren Knoten Zufallsvariablen und deren Kanten Abhängigkeiten zwischen eben diesen Variablen darstellen.

Während der Vorhersage wird sie benutzt, um die bestmögliche Sequenz $\hat{y}$ zu finden: Über alle möglichen Sequenzen von Labels hinweg wird diejenige ausgewählt, die die bedingte Wahrscheinlichkeit maximiert. $Z(x)$ stellt den Normalisierung-Faktor dar [80].

Beispiele für Feature-Funktionen

Für ein POS Tagging (siehe 2.2.3) Problem kann man in einer reduzierten Form T auf die Menge $\{SUBSTANTIV, VERB, PRPOSITION, KONJUNKTION\}$ sowie S als Menge aller deutschen Wörter setzen, falls man jeden Satz als eine Sequenz von Wörtern betrachtet. Nun ist es möglich, Feature-Funktionen für Sequenzen der Länge n auf diesen Mengen zu definieren:

$$f_0(y,x,t) = \begin{cases} 1 & \text{falls } t = 0 \wedge y_t = \text{VERB} \wedge x_{n-1} = \text{'?'} \\ 0 & \text{sonst} \end{cases} \tag{26}$$

Diese Funktion ist also 'aktiv', falls das momentane Wort der Beginn der Satzes ist, es ein Verb darstellt und der Satz mit einem Fragezeichen endet. Das ist normalerweise bei einer Frage der Fall, wie zum Beispiel 'Sollte dies anonymisiert werden?'. Intuitiv sollte diese Funktion nun ein hohes, positives Gewicht λ_0 erhalten, denn viele Fragen beginnen mit einem Verb - daher sollte LCRF Sequenzen von Labels y bevorzugen, welche dem ersten Wort das Label 'VERB' zuweisen.

$$f_1(y,x,t) = \begin{cases} 1 & \text{falls } y_t = \text{KONJUNKTION} \wedge y_{t-1} = \text{KONJUNKTION} \\ 0 & \text{sonst} \end{cases} \tag{27}$$

Diese Funktion hingegen ist 'aktiv', falls sowohl das momentane als auch das vorhergegangene Wort das Label 'KONJUNKTION' erhalten soll. Da in der Regel keine zwei Konjunktionen direkt aufeinanderfolgen, sollte das zugehörige Gewicht λ_0 nun einen stark negativen Wert erhalten, um solche Labelings zu vermeiden.
Die Erstellung solcher Feature-Funktionen ist zu einem überwiegenden Teil Handarbeit und meist relativ aufwendig, da gute LCRF Systeme in der Regel recht komplex sind und relativ viele solcher Funktionen nutzen [80].

Erlernen der Gewichte

Zur Durchführung des Trainings mithilfe der Daten (X, Y) wird das Maximum-Likelihood [10] Verfahren angewendet. Dafür betrachten wir die Sequenzen $X = (x^{\{0\}}, ..., x^{\{I-1\}})$ mit den dazugehörigen Labels $Y = (y^{\{0\}}, ..., y^{\{I-1\}})$, wobei $I \in \mathbb{N}$ die Anzahl der vorhandenen Beispiele notiert.

$$L(\lambda) = \prod_{i=0}^{I-1} P(y^{\{i\}} | x^{\{i\}}, \lambda) \tag{28}$$

Auf Basis des Verfahrens gilt es nun, die Likelihood $L(\lambda)$ zu maximieren. Äquivalent ist es, die Log-Likelihood zu maximieren:

$$\begin{aligned} LL(\lambda) &= \sum_{i=0}^{I-1} log(P(y^{\{i\}} | x^{\{i\}}, \lambda)) \\ &= \sum_{i=0}^{I-1} log(\frac{1}{Z(x^{\{i\}})} \cdot exp(\sum_{t=0}^{n-1} \sum_{k=0}^{K-1} \lambda_k \cdot f_k(y^{\{i\}}, x^{\{i\}}, t))) \\ &= \sum_{i=0}^{I-1} \sum_{t=0}^{n-1} \sum_{k=0}^{K-1} \lambda_k \cdot f_k(y^{\{i\}}, x^{\{i\}}, t) - \sum_{i=0}^{I-1} log(Z(x^{\{i\}})) \end{aligned} \tag{29}$$

[10] Maximum-Likelihood ist ein Schätzverfahren für Parameter, um denjenigen Wert für Parameter auszuwählen, welcher für gegebene Daten deren Realisierung am Wahrscheinlichsten macht. Es wird derjenige Parameter ausgewählt, welcher die sogenannte Likelihood-Funktion maximiert.

Ein Weg, um die Gefahr von Overfitting zu verringern, ist der Einsatz eines Regulierungs-Terms - er 'bestraft' zu große Werte für die Gewichte λ_k. In diesem Fall wird eine quadratische Regulierung $\sum_{k=0}^{K-1} \lambda_k^2$ gegenüber einer betraglichen $\sum_{k=0}^{K-1} |\lambda_k|$ bevorzugt, da ersterer in 0 differenzierbar ist und somit die Optimierung vereinfacht. Der Term wird mit $\frac{1}{2 \cdot \sigma^2}$ gewichtet, sodass mithilfe verschiedener Werte von σ^2 der Einfluss des Regelarisierungs-Terms angepasst werden kann. Die optimalen Gewichte $\hat{\lambda}$ ergeben sich aus der Maximierung des regulierten Terms:

$$LL(\lambda) = \sum_{i=0}^{I-1} \sum_{t=0}^{n-1} \sum_{k=0}^{K-1} \lambda_k \cdot f_k(y^{\{i\}}, x^{\{i\}}, t) - \sum_{i=0}^{I-1} log(Z(x^{\{i\}})) - \sum_{k=0}^{K-1} \frac{\lambda_k^2}{2 \cdot \sigma^2} \tag{30}$$
$$\hat{\lambda} = argmax_\lambda LL(\lambda)$$

Da es nicht möglich ist, diese Funktion analytisch zu maximieren, werden in der Regel numerische Optimierer eingesetzt, um $\hat{\lambda}$ zu bestimmen [80].

2.3.9 Deep Learning

Deep Learning beschreibt eine Gruppe von Ansätzen innerhalb des Maschinellen Lernens und wird häufig in Zusammenhang mit neuronalen Netzwerken gesehen. Goodfellow et al. legen dar, dass sich Ansätze dieser Gruppe vor allem durch tiefe, hierarchische Strukturen auszeichnen, wie es eben zum Beispiel bei tiefen, neuronalen Netzwerken der Fall ist. Dies trifft aber durchaus auch auf andere, zum Beispiel graphische, Modelle zu [33]. Diese Definition unterstützen auch LeCun et al.: "A deep-learning architecture is a multilayer stack of simple modules", wobei ihre Definition den Terminus der 'einfachen Module' beinhaltet. Diese wären, zum Beispiel im Falle von Neuralen Netzwerken, einzelne Schichten verschiedener Arten (Rekurrent, Convolution, Vollständig-Verbunden...) [48]. Insbesondere die in dieser Arbeit verwendeten RNNs werden zu Deep Learning Techniken gezählt, da sie durch die Abhängigkeiten von vorhergehenden (beziehungsweise auch folgendenden, im Falle von BRNNs) Eingaben sehr tiefe Strukturen bilden. Auch die verwendeten CNNs werden dazu gezählt - denn diese setzen meist auf mehre Abfolgen von Convolution- sowie Pooling-Schichten [73].
Deep Learning Ansätze haben sich in vielen verschiedenen Bereichen durchgesetzt, da sie deutlich bessere Ergebnisse als alternative Ansätze erreichen [73]. Als besonders vorteilhaft wird unter anderem gesehen, dass Deep Learning Ansätze kaum Feature Enginneering benötigen und auch mit guten Ergebnissen auf rauschenden Daten eingesetzt werden können [48].

2.3.10 Evaluation eines Modells

Ein ungemein wichtiger Faktor im maschinellen Lernen ist die Evaluation von Modellen: Denn egal wie gut sie sind, jedes ML-Modell macht in der Regel Fehler. Dies verhält sich sehr ähnlich wie bei Werksmaschinen in in einer Produktion: Auch sie machen gewisse Fehler, diese werden meist in Toleranzen angegeben, die sich zum Beispiel auf die Genauigkeit beim Schneiden eines Materials beziehen. Solche Angaben sind sehr wichtig, da dem Betreiber der Maschine bekannt ist, bis zu welchem Grad er sich auf sie verlassen kann. Im Bereich des maschinellen Lernens gibt man daher entsprechende Messgrößen an, welche den zu erwartenden Fehler abschätzen. Wie in Sektion 2.3.3 erläutert, sind dabei für die jeweiligen Anwendungen (Training, Entwicklung beziehungsweise Tests) die korrekten Datensätze einzusetzen. Die in dieser Arbeit in Sektion 4 verwendeten Messgrößen werden in den folgenden Paragrafen zuerst für den Fall einer binären Klassifikation erläutert und dann die Erweiterung auf den Multiklassenfall beschrieben.

Konfusionsmatrix

Im Falle einer binären Klassifikation gibt es genau 2 Formen, die ein Ergebnis annehmen kann: Positiv (1) sowie Negativ (0). Dies gilt natürlich sowohl für die Labels (das gewünschte Ergebnis) als auch die Vorhersage des Systems. Stellt man nun alle Ergebnisse in einer Tabelle dar, sieht das wie folgt aus:

	Klasse: 1	Klasse: 0	
Vorhersage: 1	richtig positiv (r_p)	falsch positiv (f_p)	$r_p + f_p$
Vorhersage: 0	falsch negativ (f_n)	richtig negativ (r_n)	$f_n + r_n$
	$r_p + f_n = P$	$f_p + r_n = N$	$P + N = E$

Tabelle 4: Konfusionsmatrix für den Fall einer binären Klassifikation

In grün markiert sind hierbei die Fälle, welche der Klassifizierer richtig beurteilt (r_p, auch true positives genannt sowie r_n, auch true negatives genannt), wobei rot die Fälle markiert, in denen die Vorhersage falsch ist (f_p, auch false positives genannt sowie f_n, auch false negatives genannt). Weiterhin bezeichnet P die Gesamtanzahl aller positiven Beispiele, N die Gesamtanzahl aller negativen Beispiele sowie E die Gesamtzahl aller Beispiele ungeachtet deren Klasse [41].
Im folgenden bezeichnen r_p, r_n, f_p sowie f_n die Anzahl der Beispiele, welche in die jeweils zugehörige Kategorie fallen. Des weiteren liegen die Ergebnisse der meisten Metriken zur Auswertung der Leistung eines Systems im Bereich $[0, 1]$, wobei ein höheres Ergebnis als besser betrachtet wird. Im Fließtext werden die Werte aber zur besseren Verständlichkeit als Prozentzahl wiedergegeben - damit entsprechen z.B. 0% einem Wert von 0,0 , 50% einem Wert von 0,5 sowie 100% einem Wert von 1.
Die Konfusionsmatrix lässt sich leicht auf beliebig viele Klassen erweitern, in dem man entsprechend viele Zeilen/Spalten hinzufügt, wie in Tabelle 5 dargestellt. Die Begrifflichkeiten der 'richtig positiven', 'falsch positiven' et cetera fallen hierbei weg.

	Klasse: A	Klasse: B	Klasse: C
Vorhersage: A	richtig	falsch	falsch
Vorhersage: B	falsch	richtig	falsch
Vorhersage: C	falsch	falsch	richtig

Tabelle 5: Konfusionsmatrix für den Fall einer Klassifikation mit 3 Klassen

Die Konfusionsmatrix ist dabei auch gut geeignet um herauszufinden, welche Klassen untereinander von einem ML-System häufig verwechselt werden [41].

Accuracy

Accuracy bezeichnet den Anteil aller korrekt klassifizierten Beispiele an der Gesamtmenge aller Beispiele. Dementsprechend lautet die Formel:

$$Accuracy = \frac{r_p + r_n}{E} \quad Accuracy \in [0, 1] \tag{31}$$

Accuracy wird oft als ein erstes Maß verwendet - denn zum einen ist es sehr einfach zu berechnen, zum anderen sehr intuitiv: Wie viele Werkstücke eine Maschine von allen Werkstücken, die man ihr übergibt, korrekt bearbeitet, ist die naheliegendste Art die Leistung der Maschine zu beurteilen. Doch gerade im Maschinellen Lernen ist dies nicht immer die beste Art, die Leistung eines ML-Modells zu messen [11]. Um dies zu veranschaulichen lässt sich der sogenannte 'Brustkrebs Datensatz' heranziehen [55]. Es ist eins der Standard-Beispiele für viele ML Probleme: Es enthält je 9 Features von 286 Frauen, welche an Brustkrebs litten, aber geheilt werden konnten. Aufgeteilt sind sie in zwei Klassen: Ob der Brustkrebs bei ihnen innerhalb von 5 Jahren wieder aufgetaucht ist (Klasse '1') oder nicht (Klasse '0'). Dieser Datensatz ist unausgewogen, das heißt das Verhältnis der beiden Klassen ist unausgeglichen: 201 gehören der Klasse '0' an, aber nur 85 der Klasse '1'.

	Klasse: '1'	Klasse: '0'	
Vorhersage: '1'	0	0	0
Vorhersage: '0'	85	201	286
	85	201	286

Tabelle 6: Konfusionsmatrix für das Beispiel der Krebsvorhersage: System sagt immer '0' vorraus

Daraus ergibt sich das sogenannte 'Accuracy Paradox': Ein Modell, welches für alle Patienten als Ergebnis '0' vorhersagt, erreicht hierbei eine relativ hohe Accuracy von 70,28% ($\frac{0+201}{286}$), ohne dass das Modell dabei etwas gelernt hat. Um solche Fehler zu entdecken, benötigt es also zusätzliche Maße, welche in den nächsten Paragraphen eingeführt werden. Wie so etwas hingegen noch während des Lernens eines ML-Modelles behandelt werden kann, wird in dem Abschnitt 'Kosten-Sensitives Lernen' thematisiert.
Eine Erweiterung auf mehrere Klassen lässt sich dadurch erreichen, dass die Anzahl aller richtig klassifizierten Beispiele über alle Klassen hinweg durch die Gesamtanzahl aller Beispiele geteilt wird.

Precision, Recall und F-Score

Als zusätzliche Maße werden meist Precision sowie Recall herangezogen. Sie berechnen sich wie folgt:

$$\begin{aligned} Precision &= \frac{r_p}{r_p + f_p} \quad Precision \in [0,1] \\ Recall &= \frac{r_p}{P} \quad Recall \in [0,1] \end{aligned} \tag{32}$$

Während Precision den Anteil der korrekten Vorhersagen über alle als positiv vorhergesagten Beispiele wiedergibt (sich damit also auf die erste Zeile der Konfusionsmatrix beschränkt), repräsentiert Recall den Anteil der richtig positiv vorhergesagten Beispiele über alle tatsächlich positiven Instanzen hinweg (bezieht sich dementsprechend auf die erste Spalte der Konfusionsmatrix). Recall wird oft auch als die Sensitivität eines ML-Modells bezeichnet: Denn umso eher (z.B. bei einem niedrigeren Schwellwert) es Beispiele als positiv klassifiziert (also sensibler reagiert), umso höher ist r_p und umso kleiner ist f_n - damit steigt auch der Recall. Precision hingegen kann man durchaus wörtlich übersetzt als Präzision bezeichnen - es sagt aus, wie viele der als positiv klassifizierten Beispiele tatsächlich positiv sind. Daher ist es auch schwer, Recall und Precision gleichzeitig zu maximieren - denn umso 'leichtfertiger' ein Beispiel als positiv klassifiziert wird (hoher Recall), desto öfter sind meist auch negative Beispiele dabei und die Precision sinkt. Bei einem hohen Recall kann man sich also sicher sein, die meisten positiven Beispiele auch als solche zu klassifizieren, man muss aber damit rechnen, auch einige negative darunter zu haben. Will man sich hingegen sicher sein, dass die meisten der als positiv klassifizierten Beispiele auch tatsächlich solche sind (hohe Präzision), muss man hingegen oft damit leben nicht alle positiven Beispiele auch zu finden. Dieser Abtausch wird auch als Precision-Recall Tradeoff bezeichnet [11].
Als Folge daraus, dass zwischen hoher Precision sowie hohem Recall ein Kompromiss gefunden werden muss, kommt die sogenannte F_β-Score (oder auch F_β-Measure genannt) ins Spiel:

$$F_\beta = (1+\beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall} \quad F_\beta \in [0,1] \tag{33}$$

Es bezieht sowohl Precision als auch Recall mit ein, β bestimmt deren Gewichtung - meist benutzt ist hierbei der F_1, in dem also $\beta = 1$ gilt und Precision sowie Recall gleichwertig gewichtet werden. Für diesen Fall lässt sich der F_1-Score auch simpel aus den Grundelementen ausrechnen:

$$F_1 = (1+1^2) \cdot \frac{precision \cdot recall}{(1^2 \cdot precision) + recall} = \frac{precision \cdot recall}{precision + recall} = \frac{2 \cdot r_p}{2 \cdot r_p + f_r + f_n} \tag{34}$$

Für das Beispiel von oben ergibt sich bei der Berechnung der Precision ein Problem - eine Division durch 0: $Precision = \frac{0}{0+0}$. Es gibt wechselnde Definitionen, welchen Wert Precision in solch einem Fall annimmt (0, 1 oder 'undefiniert'). Anhand der oben genannten Intuition hinter Precision sowie dem Gedanken hinter dem Precision-Recall Tradeoff, wonach bei einem niedrigen Recall in der Regel die

Precision hoch ist, wird in dieser Arbeit die Definition $Precision = \frac{0}{0+0} \hat{=} 1$ verwendet.
Damit ergeben sich für das Beispiel die folgenden Werte:

$$\begin{aligned} Precision &= \frac{0}{0+0} \hat{=} 1 \\ Recall &= \frac{0}{85} = 0 \\ F_1 &= (1+1^2) \cdot \frac{1 \cdot 0}{1+0} = 0 \end{aligned} \tag{35}$$

Auf den ersten Blick wirkt es, als wäre der schlechte Klassifizierer entlarvt: Sowohl Recall als auch F_1-Score sind an ihrem Minimum angelangt. Doch der Schein trügt, wie folgendes Experiment zeigt: Dreht man die Verteilung der Klassen um (ändert also die Labels von '0' zu '1' sowie umgekehrt) und lässt den Klassifizierer nun immer '1' Voraussagen (dies ist eine legitime Herangehensweise, da dem Entwickler jederzeit frei steht, welche Klassen er als 'positiv' beziehungsweise 'negativ' bezeichnet), ändert sich die Konfusionsmatrix wie folgt:

	Klasse: '1'	Klasse: '0'	
Vorhersage: '1'	201	85	286
Vorhersage: '0'	0	0	0
	201	85	286

Tabelle 7: Konfusionsmatrix für das umgedrehte Beispiel der Krebsvorhersage: System sagt immer '0' vorraus

Und für die Maße ergibt sich damit das folgende:

$$\begin{aligned} Accuracy &= \frac{201+0}{286} = 0,70 \\ Precision &= \frac{201}{201+85} = 0,70 \\ Recall &= \frac{201}{201+0} = 1 \\ F_1 &= 2 \cdot \frac{0,7 \cdot 1}{0,7+1} = 0,82 \end{aligned} \tag{36}$$

Während die Accuracy gleich bleibt, steigen Precision, Recall sowie der F_1-Score auf hohe Werte an. Dies ist dem Aufbau der Formeln zu schulden: Im Zähler stehen bei Precision sowie Recall ausschließlich r_p und die Formeln berücksichtigen jeweils nur zwei der vier Felder der Konfusionsmatrix - auch mit dem F_1-Score werden nur drei der Felder abgedeckt (r_n wird nicht berücksichtigt). Solange ein Klassifizierer sehr gute Ergebnisse auf den positiven Beispielen erzielt, wird er also gute Ergebnisse in diesen Maßen erhalten. Accuracy, Precision, Recall sowie der F_β-Score können also bei einem unausgewogenen Datensatz missverständliche Aussagen generieren.
Für die Anwendung im Falle der Multiklassifikation existieren 2 verschiedene Arten, Precision, Recall sowie $F_1 - Score$ zu berechnen, wobei die Methoden unterschiedliche Werte ergeben können:

Macro-Average Die jeweilige Metrik wird für jede Klasse getrennt berechnet und dann durch die Berechnung des Durchschnittes zu einem einzelnen Wert zusammen gefasst - dadurch wird jede Klasse, unabhängig ihrer Häufigkeit des Aufkommens, äquivalent gewichtet

Micro-Average r_p, r_n, f_p sowie f_n werden über alle Klassen hinweg aufaddiert und die Metrik dann entsprechend ihrer Formel berechnet - dadurch wird jede Klasse, abhängig von der Häufigkeit ihres Aufkommens, gewichtet

Als falsch positive zählen dabei die Werte aus der zu der jeweiligen Klasse gehörigen Zeile (ausgenommen der Diagonalen), als falsche negative die Werte aus der zugehörigen Spalte (ausgenommen der Diagonalen).

Matthews Correlation Coefficient (MCC)

1975 von B.W.Matthews eingeführt, bildet der Matthews Correlation Coefficient (MCC) eine Alternative zu den oben erklärten Maßen [56]. Er wurde vom Pearsons Correlation Coefficient abgeleitet. Es bezieht alle Felder der Konfusionsmatrix in seiner Berechnung mit ein und erreicht so, dass hohe Ergebnisse nur von Klassifizieren erreicht werden können, die sowohl negative als auch positive Beispiele gut klassifizieren [16]. Berechnet wird er wie folgt, wobei -1 die schlechtmöglichste Klassifikation darstellt, 1 die bestmögliche:

$$MCC = \frac{(r_p \cdot r_n) - (f_p \cdot f_n)}{\sqrt{(r_p + f_p) \cdot (r_p + f_n) \cdot (r_n + f_p) \cdot (r_n + f_n)}} \quad MCC \in [-1, 1] \tag{37}$$

Für Fälle, in welchem der Nenner 0 entspricht und der Wert undefiniert wäre, wird er auf 1 gesetzt - dies ist Konvention und rührt daher, dass auf diese Weise die Grenzwerte übereinstimmen.
Die Berechnung ist komplexer als bei den oben genannten Maßen - setzt man nun die Werte für unsere beiden Beispiele ein, erhält man:

$$\begin{aligned} MCC &= \frac{(0 \cdot 201) - (0 \cdot 85)}{\sqrt{(0+0) \cdot (0+85) \cdot (201+0) \cdot (201+85)}} = \frac{0}{0} \hat{=} 0 \\ MCC &= \frac{(201 \cdot 0) - (85 \cdot 0)}{\sqrt{(201+85) \cdot (201+0) \cdot (0+85) \cdot (0+0)}} = \frac{0}{0} \hat{=} 0 \end{aligned} \tag{38}$$

MCC zeigt also bei beiden Fällen korrekt die mangelhafte Klassifikation an. Außerdem hat Davide Chicco gezeigt, dass auch bei weniger extremen Fällen, in welchen Precision, Recall und F_1-Score irreführend sein können, der MCC angemessene Werte erzeugt [16].
Für die Anwendung in einer Multiklassen-Klassifikation mit K-Klassen wurde MCC explizit erweitert. Für eine Konfusionsmatrix C, auf dessen Felder C_{lm} mithilfe der Indices l sowie m zugegriffen werden kann, berechnet sich MCC wie folgt:

$$MCC = \frac{\sum_k^K \sum_l^K \sum_m^K C_{kk} C_{lm} - C_{kl} C_{mk}}{\sqrt{\sum_k^K \left(\sum_l^K C_{kl}\right) \left(\sum_{k'|k' \neq k}^K \sum_{l'}^K C_{k'l'}\right)} \sqrt{\sum_k^K \left(\sum_l^K C_{lk}\right) \left(\sum_{k'|k' \neq k}^K \sum_{l'}^K C_{t'k'}\right)}} \tag{39}$$

Es ist auch möglich, MCC mithilfe von Micro,-beziehungsweise Macro-Averaging (siehe oben) auf den Multiklassen-Fall zu erweitern. Doch in diversen Anwendungsfällen hat sich die Verwendung der, in Berechnung 39 gegebene, Formel als vorteilhaft gezeigt [43].
Häufig wird MCC im maschinellen Lernen in Verbindung mit Medizin eingesetzt, wo ungleichmäßige Klassenverteilungen häufiger sind als in anderen Bereichen (wie z.B. in dem Brustkrebsdatensatz) - denn gerade dort ist seine Verwendung wie gezeigt vorteilhaft (Dies ist bei einer Multiklassen-Klassifikation nur noch abgeschwächt der Fall). [12] [43]. Gerade dies macht ihn auch für die Anonymisierungs-Aufgaben dieser Arbeit geeignet: Denn dadurch, dass die größten Teile eines Textes meistens nicht anonymisiert werden müssen (alleine schon durch Verbindungswörter, Pronomen etc.), können Systeme auch hier mit der einfachen Vorhersage 'Nicht zu anonymisieren' eine hohe Accuracy erreichen. Daher wird dieses Maß (neben der $F_1 - Score$) eine wichtige Rolle in der Evaluation dieser Arbeit einnehmen, insbesondere bei der binären Auswertung (siehe 4.4).

Receiver Operator Characteristic (ROC)

Einen grafischen Weg, ein ML-Modell zu evaluieren, bietet die Receiver Operator Characteristic (ROC) - in einem 2D-Graphen werden auf der Y-Achse Recall (auch True-Positive Rate (TPR) genannt) und auf der X-Achse die False-Positive Rate (FPR) aufgeführt. Diese berechnen sich wie folgt:

$$\begin{aligned} TPR = Recall &= \frac{r_p}{P} \quad TPR \in [0, 1] \\ FPR &= \frac{f_p}{N} \quad FPR \in [0, 1] \end{aligned} \tag{40}$$

Anhand dieser Werte können dann Klassifizierer in dem gewonnenen Koordinatensystem (auch ROC-Space genannt) eingetragen werden. Dies können zum einen konzeptionell unterschiedliche Systeme, zum anderen aber auch baugleiche Systeme mit unterschiedlichen Hyperparametern sein - daher eignet sich diese Darstellung auch zur Optimierung von Hyperparametern. Solch ein ROC-Space ist beispielhaft in Abbildung 15 dargestellt.

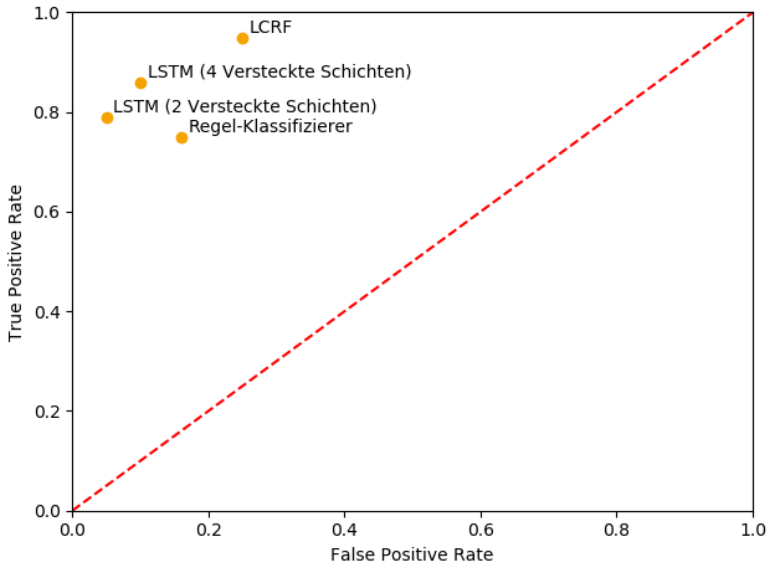


Abbildung 15: Eine Beispielhafte Darstellung für die Leistungen verschiedener Klassifizierer im ROC-Space

Die rote Linie stellt die Diagonale dar, auf welcher $TPR = FPR$ gilt. Liegt ein Klassifizierer auf dieser Linie, ist dessen Leistung nicht besser als ein (an die Häufigkeiten von positiven und negativen Beispielen angeglichenes) "Raten", da in diesem Fall anteilig von allen positiven Beispielen genau so viele richtig klassifiziert werden, wie von den negativen Beispielen falsch. Ein Klassifizierer sollte also in der Regel oberhalb der Diagonale liegen.

Optimierung im ROC-Space: Der ROC-Space kann auch verwendet werden, um einen optimalen Klassifizierer zu finden: Dazu 'schiebt' man gedanklich eine Diagonale mit der Steigung $r = \frac{N}{P}$ 'von oben' in den ROC-Space hinein. Den Klassifizierer (gegebenenfalls auch mehrere), der dabei als Erstes von der Diagonale getroffen wird, ist dabei der optimale Klassifizierer für dieses Verhältnis von Positiven zu Negativen Beispielen. Falls negative beziehungsweise positive Beispiele verschieden gewichtet werden sollen, lässt sich dies mit einer Anpassung der Steigung durchführen - für eine doppelte Gewichtung von positiven Beispielen würde sich die Steigung dementsprechend wie folgt berechnen: $r = \frac{N}{2 \cdot P}$ (näheres dazu in Sektion 2.3.6). Ein Beispiel dazu ist in Abbildung 16 gegeben.

Ranker im ROC-Space: Es existieren auch Klassifizierer, die als Vorhersage nicht nur eine Klasse, sondern ein 'Ranking' ausgegeben - so wird zum Beispiel für jeden Datenpunkt, der in diesen Klassifizierer gegeben wird, ein Wert zwischen 0 und 1 ausgegeben. Hierbei werden Beispiele mit höheren Werten (welche näher an 1 liegen) von dem Klassifizierer als zugehöriger zur Klasse 1 angesehen als Beispiele mit niedrigeren Werten. Setzt man nun einen Schwellenwert, kann man dieses 'Ranking' in eine Klassifizierung umwandeln: Alle Werte unterhalb oder gleich des Schwellwertes werden als '0' klassifiziert, alle Werte größer dem Schwellwert werden als '1' klassifiziert. Dies kann zum Beispiel für die Klassifikation mithilfe eines neuronalen Netzes eingesetzt werden - in diesem Rahmen wird es unter anderem auch in der Evaluation in dieser Arbeit eine wichtige Rolle spielen.
Auch hier ist oft das Problem, einen optimalen Schwellenwert zu finden - auch hierfür kann man gut den ROC-Space verwenden: Anfangend bei einem Schwellenwert von 1 (alle Beispiele werden als '0' klassifiziert, Punkt im ROC-Space: (0, 0)), verringert man diesen Wert in möglichst kleinen Schritten bis zu einem Wert von 0 (alle Beispiele werden als '1' klassifiziert, Punkt im ROC-Space: (1, 1)). Für

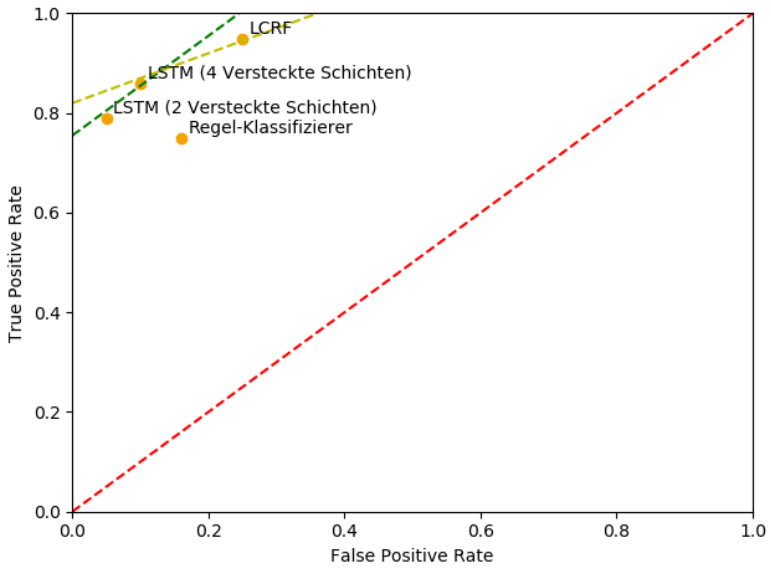


Abbildung 16: Ein Beispiel, wie man mithilfe von Diagonalen die besten Klassifizierer bestimmen kann - die grüne Linie besitzt eine Steigung von $r = 1$, die gelbe von $r = \frac{1}{2}$. Für das erstere Kostenverhältnis ist das LSTM mit 4 versteckten Schichten (vergleiche Sektion 2.3.7) der optimale Klassifizierer, für zweitere das LCRF (vergleiche Sektion 2.3.8)

jeden dieser Zwischenschritte rechnet man nun die TPR sowie FPR Werte aus - diese Werte lassen sich nun zu einer Kurve verbinden, wie Abbildung 17 zeigt. Jede dieser Kurven beginnt in dem Punkt (0, 0) und endet in dem Punkt (1, 1). Während sich die oben genannten Techniken zur Optimierung auch für Kurven im ROC-Space anwenden lassen, ist es auch möglich, weitere Leistungs-Aussagen zu treffen: Hat die Linie eines Rankers lokal die Steigung 1 (verläuft also Diagonal) oder sogar eine geringere, ist der Klassifizierer lokal nicht besser, beziehungsweise schlechter, als eine zufällige Klassifikation, da die FPR mindestens genauso schnell steigt wie die TPR.

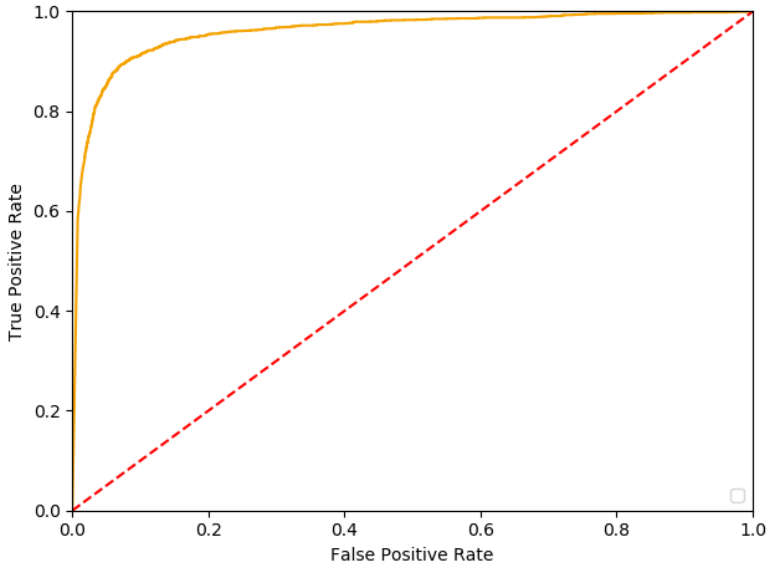


Abbildung 17: Beispiel für eine ROC-Kurve

Dementsprechend ist auch eine konvexe Kurvenform einer konkarven sowohl lokal als auch global zu bevorzugen (Dementsprechend würde man die Form der Kurve aus Abbildung 17 als 'gut' bezeichnen). Ein weiteres Kriterium, um die Performance zu messen, ist die 'Area under Curve' (AUC), welches die Fläche unter einer Kurve misst - dieses Kriterium gilt es zu maximieren. Nimmt man alles zusammen, kann man die theoretisch optimale Kurve beschreiben: Sie steigt, von dem Punkt (0, 0) beginnend, senkrecht an bis zu dem Punkt (0, 1) - dann verläuft sie waagerecht bis zu (1, 1) - würde man den Schwellenwert des Punktes (0, 1) verwenden, würde der Klassifizierer alle Beispiele richtig klassifizieren. In der Regel ist diese Kurve aber nicht zu erreichen.

Erweiterung auf ein Multiklassenproblem: Eine Erweiterung des ROC-Space für Multiklassenprobleme lässt sich durch Micro,-sowie Macro-Averaging gewinnen, wie es in Precision, Recall und F-Score vorgestellt wurde. Es existieren noch weitere, ROC-spezifische Generalisierungen, zum Beispiel von Hand et al. 2001 [35] oder von Langrebe et al. 2007 [46] - doch diese übersteigen in ihrer Komplexität den Rahmen, in dem ROC-Charakterisitiken in dieser Arbeit eingesetzt werden.

2.4 Zusammenfassung

In dieser Sektion wurde zu Beginn herausgearbeitet, welche rechtlichen Grundlagen die Anonymisierung von Texten notwendig machen (Sektion 2.1). Dies führt, auf Basis der Europäische Datenschutz-Grundverordnung (DSGVO), auf den Schutz personenbezogener Daten zurück. Es existiert dabei keine vollständige Liste, welche alle Arten von Daten beinhaltet, die anonymisiert werden müssen. Vielmehr ist dies vom jeweilige Kontext anhängig.
Grundlegend für die Durchführung von Anonymisierungen ist die Fähigkeit, natürliche Sprache zu verarbeiten. Daher wurden in Sektion 2.2 verschiedene Methoden zur Verarbeitung natürlicher Sprache, wie zum Beispiel Embeddings, eingeführt. Diese dienen als Basis für Methodiken, die die tatsächliche Anonymisierung durchführen. Als eben solche wurden verschiedene Techniken des ML, zusammen mit grundlegendem Wissen aus diesem Bereich, eingeführt (Sektion 2.3). Herauszuheben sind hierbei neuronale Netzwerke sowie Conditional Random Fields, da sie essentieller Teil dieser Arbeit sind. Da Methoden des ML nicht fehlerfrei arbeiten, wurden des weiteren Maße wie der F1-Score vorgestellt, mit dessen Hilfe die Leistungen solcher Methoden beurteilt werden können (Sektion 2.3.10). In der nächsten Sektion werden diese im Besonderen eine wichtige Rolle im Vergleich von verwandten Arbeiten einnehmen.

3 Automatische Methoden zur Anonymisierung von Texten

Der Großteil der Bereiche, die in dieser Arbeit abgedeckt werden, genießen regelmäßige sowie aktuelle Forschung. Da eine Vielzahl an Arbeiten existiert und sich gerade im Feld des Maschinellen Lernens viel an der eingesetzten Methodik in den letzten Jahren geändert hat (wie diese Sektion zeigen wird), werden in diesem Kapitel eine Auswahl der aktuellsten und für diese Arbeit relevantesten Veröffentlichungen aus den jeweiligen Bereichen erläutert und analysiert. Die Meisten von ihnen beschäftigen sich mit der Anonymisierung beziehungsweise NER in Englischer Sprache. Nur einige von ihnen liefern auch Vergleichsergebnisse für deutsche Texte. Die Ergebnisse lassen sich in der Regel über verschiedene Sprachen hinweg übertragen. Daher können die Erkentnise über die relative Leistung verschiedener Methodiken aus der Englischen in die deutsche Sprache übertragen werden, wie einige Arbeiten zeigen [29] [32] [25].

3.1 Klassische Methoden

In der Industrie, in welcher Anonymisierung im kommerziellen Rahmen eingesetzt wird, werden (abseits der medizinischen Anwendung) häufig Methoden eingesetzt, die nicht auf Maschinellem Lernen basieren, wie in Sektion 3.2.1 näher beschrieben wird. Da diese Methodiken sowohl in der Anonymisierung (im medizinischen Bereich) als auch in dem Verwandten Bereich der NER mit der Zeit durch Methoden des ML abgelöst wurden, werden diese im Rahmen dieser Arbeit als klassische Methoden bezeichnet (vergleiche Sektionen 3.2.1 sowie 3.2.2). In dieser Sektion werden daher zuerst grundlegende, klassische Methodiken erläutert, bevor dann das Vergleichssystem aus der Industrie sowie eine weitere Arbeit aus diesem Bereich näher erläutert werden.

Reguläre Ausdrücke

Reguläre Ausdrücke sind Zeichenketten, mit dessen Hilfe Texte nach Zeichenfolgen durchsucht werden können, die einem, durch den Regulären Ausdruck definierten Muster, folgen. So können solche Ausdrücke zum Beispiel Telefonnummern oder E-Mails identifizieren. Diese Muster werden zum einen durch simple, alphabetische Zeichen definiert, zum anderen durch spezielle Steuerzeichen, die komplexe Ausdrücke zulassen. Im folgenden wird dies anhand des Beispieles von Regulären Ausdrücken in Python [11] erläutert. Bei Regulären Ausdrücken existieren Steuerzeichen, welche Wiederholungen von Buchstaben in beliebiger Menge erlauben ('*'), oder Gruppen von Zeichen definieren, welche jeweils an dieser Stelle vorkommen können ('[]'). So würde der Reguläre Ausdruck 'B[a,ae,ä]r' unter anderem 'Bar', 'Bär' sowie 'Baer' matchen, aber auch 'Br', 'Bääääääär' oder 'Baär' finden. Weitere Informationen sowie einen Überblick über die verfügbaren Steuerzeichen in Python sind unter `https://docs.python.org/2/library/re.html` zu finden.
In dieser Arbeit werden sie, neben dem Vergleichsystem aus der Industrie, auch im Rahmen der Vervollständigung des Dortmund Chat Korpus eingesetzt (vergleiche Sektion 4.1.1).

Nachschlagetabellen

Nachschlagetabellen, im Englischen Lookup-Tables genannt, werden genutzt, um Informationen über bekannte Entitäten abzulegen und verschiedenen Verfahren, zum Beispiel als Feature, zugänglich zu machen. So werden Nachschlagetabellen häufig in der NER eingesetzt, wo sie zum Beispiel für einen String Auskunft darüber geben, ob dieser in einer Liste von bekannten Namen oder Orten enthalten ist [64]. Eine größere Sammlung an Nachschlagetabellen nennt man auch Wissensdatenbank oder Knowledge-Base [17].

Neamatullah et al.

Zum Vergleich für klassische Ansätze in der Anonymisierung von medizinischen Daten, lässt sich die Arbeit von Neamatullah et al. von 2008 heran ziehen. Sie führten die Anonymisierung mittels Regulärer

[11] Python, Programmiersprache. Mehr Informationen unter `https://www.python.org/`

Ausdrücke, Nachschlagetabellen für einzelne Wörter sowie einiger Heuristiken durch. So erreichten sie auf einem medizinischen Datensatz mit rund 1800 zu identifizierenden Entitäten (hauptsächlich Namen, Daten, sowie Orte Telefonnummern) einen F1-Score von 72,42% einen Recall von 96,70% sowie 74,90% Precision [62]. Dies sind, im Vergleich zu Ergebnissen verschiedener ML-Systeme, welche in Sektion 3.2.1 vorgestellt werden, niedrige Werte.

Vergleichssystem aus der Industrie

Zum Vergleich liegt für diese Arbeit ein System aus der Industrie vor, welches dort im laufenden Betrieb zur Anonymisierung eingesetzt wird [12]. Es handelt sich dabei um ein System, welches kein Maschinelles Lernen zur Erkennung einsetzt - stattdessen setzt es auf eine Kombination aus mehreren klassischen Methoden:

Nachschlagetabellen Das System nutzt Nachschlagetabellen. Diese beinhalten zum einen Wörter, die es zu anonymisieren gilt (Blacklist) als auch Wörter, welche als unbedenklich gelten (Whitelist). Diese Listen werden dann bei der Anonymisierung des Textes abgefragt.

Reguläre Ausdrücke Reguläre Ausdrücke werden eingesetzt, um zu anonymisierende Entitäten anhand bestimmter Muster zu identifizieren, zum Beispiel für Telefon- oder Kreditkartennummern.

Stichphrasen Das System nutzt Stichphrasen, um davorstehende oder darauffolgende Wörter anhand deren Kontext identifizieren zu können. Ein Beispiel für eine Stichphrase wäre 'Sehr geehrte Frau', da es sich bei den nachfolgenden Wörtern mit hoher Wahrscheinlichkeit um Namen handelt, welche es zu anonymisieren gilt.

Nach der Anwendung der Methoden gibt das System den Originaltext zurück, in welchem die zu anonymisierende Entitäten durch Platzhalter mit ihrem entsprechendem Typen versehen sind, zum Beispiel 'FullName' oder 'CreditCardNumber'.
Im restlichen Verlauf wird das System als KSystem (Kontroll-System) referenziert.

3.2 Maschinelles Lernen

Im Bereich der Anonymisierung, als auch im Bereich der NER, werden die besten Ergebnisse der letzten Jahre von ML-Systemen erreicht, wie sich in diesem Abschnitt zeigen wird. Der Fokus dieser Arbeit wird daher auf diese Ansätze gelegt.

3.2.1 Anonymisierung

Im Bereich der Anonymisierung mit Maschinellem Lernen steht das Gesundheitswesens im Fokus: Es werden regelmäßig Arbeiten veröffentlicht, die sich mit der Anonymisierung von Patientendaten beschäftigen - zeitgemäße Arbeiten aus anderen Domänen, mit signifikantem Bezug zu dieser Arbeit, waren hingegen nicht aufzufinden. Des weiteren beschäftigen sich alle aktuellen, relevanten Arbeiten mit dem i2b2 Datensatz von 2014. Dieser beinhaltet 1304 Patientenakten, in welchen zu anonymisierende Entitäten mit einer von 30 Kategorien (welche wiederum in 7 Überkategorien zusammen gefasst sind) annotiert sind. Im Allgemeinen handelt es sich bei Patientenakten um reguläre Daten, welche einer festen Struktur folgen und selten Rechtschreib- oder Grammatikfehler aufweisen (siehe Sektion 2.3.3), wodurch sich die Erkenntnisse nur zu einem gewissen Teil auf diese Arbeit übertragen lassen [79].
Eine ältere Arbeit, "Evaluating the State-of-the-Art in Automatic De-identification" von Uzuner et al., ist trotz ihres Alters von 11 Jahren betrachtenswert - denn sie wird in den behandelten Arbeiten häufig

[12] Aus rechtlichen Gründen dürfen weder der Name des Systems, noch die Unternehmen genannt werden, in welchen es eingesetzt wird. Es liegen daher auch keine Messwerte bezüglich der Leistungen vor (diese werden in Sektion 4 erarbeitet).

zitiert [83]. Sie beschäftigte sich mit einer älteren Version des i2b2 Datensatzes, welche nur 8 Kategorien unterscheidet. Die besten Ergebnisse in ihren Tests lieferten statistische Lerner [13] in Kombination mit Regel-Vorlagen als Features, gefolgt von Hybriden Systemen, welche sowohl Maschinelles Lernen als auch Regeln anwenden. Darauf folgten reine ML- sowie Regel-Systeme. Doch gerade im Bereich ML hat sich seit der Veröffentlichung dieses Papers einiges getan, wie im Verlaufe dieser Sektion erläutert wird. Im Allgemeinen waren die Ergebnisse in dieser Arbeit vergleichsweise hoch, es erreichten in allen Szenarien mehrere Systeme F1-Scores von 95%. Probleme hatten viele Systeme hingegen mit außergewöhnlichen Namen, die nicht in den Trainingsdaten aufgetaucht sind, sowie mit Datumsangaben und nicht-trivialen Orten, wie zum Beispiel 'Port Authorities' [83].

I2B2 Datensatz 2014

2015 haben Chen et al. [15] einen Ansatz vorgestellt, welcher auf der Basis von HMMs [14] sowie Dirichlet Prozessen [15] als Bayessche Prior auf medizinischen Patientendaten die Klassifizierung der Tokens vornimmt. Mit verschiedenen Alternationen ihres Ansatzes erreichten sie einen F1-Score zwischen 91% und 93%. Die Autoren heben heraus, dass dieser Ansatz die Arbeit des aufgabenspezifischen Feature Engineering und Feature Learnings gegenüber vergleichbaren Ansätzen, die auf LCRFs basieren, deutlich reduziert und doch vergleichbare Ergebnisse erreicht.
Liu et al. 2015 [51] setzten eine Kombination aus zwei verschiedenen LCRFs sowie einem auf Regeln basierten Klassifizierer ein. Eines der LCRFs operierte hierbei auf Tokens, während das andere den Text auf Basis einzelner Buchstaben bearbeitete. Mit ihrem System erreichten sie einen F1-Score von 94,64%.
Ende 2016 veröffentlichten Dernoncourt et al. einen Ansatz, welcher die Anonymisierung mithilfe von RNNs vornimmt [23]. Dafür verwenden sie bidirektionale LSTM-Layers sowie GloVe [16] Word- Embeddings. Hierbei erreichten sie auf dem selben Datensatz einen F1-Score von 97,85% - ≈5% mehr als Chen et al. sowie ≈3% mehr als Lui et al. 2015. Einen weiteren, zum Vergleich herangezogenen Ansatz mit CRFs schlagen sie um ≈0,3% - eine Kombination des RNNs mit dem CRF-System liefert noch einmal leichte Verbesserungen. Bei der detaillierten Analyse stellen sie fest, dass das CRF-System zwar besser Tokens mit fester Struktur wie zum Beispiel IDs (z.B. 38:Z8912708G) erkennt, das neuronale Netz hingegen meist besser generalisiert. Dies zeigt sich zum Beispiel in der Kategorie 'Berufe', in welcher fast alle Entitäten sehr ähnliche Embeddings besitzen und somit sehr gut vom neuronalen Netz erkannt werden. Beide Ansätze zeigten jedoch Probleme mit der Erkennung von Orten, manchen Abkürzungen sowie mehrdeutigen Wörtern. Als großen Vorteil sehen die Autoren, dass ihr neuronales Netz kein Feature Engineering und Feature Learning benötigte.
Liu et al. zeigen 2017 hingegen mit einem sehr ähnlichen, auch auf bidirektionalen LSTMs basierenden Ansatz, dass das Hinzufügen einiger zusätzlicher, spezieller Features die Ergebnisse verbessern kann [52] - sie sehen Feature Engineering und Feature Learning als eine wertvolle Möglichkeit die Leistung eines neuronalen Netzwerkes im Bereich der Anonymisierung zu verbessern. Sie erreichen so einen um 0,5% besseren F1-Score auf dem i2b2 Datensatz als Dernoncourt et al.. Im Allgemeinen bestätigen Liu et al. aber deren Ergenisse und stellen heraus, dass neuronale Netze zwar kaum mehr Precision als CRFs erreichen (teilweise sogar etwas weniger), dafür aber eine deutliche Verbesserung des Recalls herbeiführen - dies ist hauptverantwortlich für die höheren F1-Scores und deckt sich auch mit den Ergebnissen von Dernoncourt et al. [52].
In seiner Master Thesis von 2016 stellt F. Dias ein Tool zur Anonymisierung von Texten verschiedener Sprachen vor, darunter auch Deutsch und Englisch. Die Kernelemente seiner Pipeline stellen zum einen die NER-Kompenente dar, zum anderen die sogenannte Second Pass Detection. Innerhalb der

[13] Als statistische Lerner werden Methoden aus dem Bereich des Maschinellen Lernens bezeichnet, welche eine enge Bindung zur Statistik aufweisen [9]

[14] Hidden Markov Modelle (HMMs) sind stochastische Modelle mithilfe welcher man ein System durch eine sogenannte Markow-Kette mit unbeobachteten Zuständen modellieren kann [44].

[15] Dietrich Prozesse beschreiben eine Familie von Stochastischen Prozessen, deren Realisierungen (also die konkreten Werte, die deren entspringen) wiederum Wahrscheinlichkeitsverteilungen darstellen.

[16] `https://nlp.stanford.edu/projects/glove/`https://nlp.stanford.edu/projects/glove/

NER-Komponente arbeiten mehrere NER-Systeme (unter anderem zum Beispiel Stanford-NER [17]) zusammen mit einem Pattern-Matcher, welcher auf Regulären Ausdrücken basiert - die finale Klassifikation wird durch ein Voting der jeweiligen Komponenten erzielt. Die Aufgabe der Second-Pass Detection ist es, Fehler in der Klassifikation der NER-Komponente zu erkennen und zu korrigieren. So macht sie auch den Unterschied zwischen NER sowie Anonymisierung aus: Bei Entitäten, welche zu anonymisieren wären, aber keine Named Entites darstellen (beziehungsweise umgekehrt), korrigiert sie das Label. Auch inkonsistente Labels, die zum Beispiel auftreten wenn das selbe Wort an zwei Stellen unterschiedliche Labels erhält, versucht sie zu erkennen. Auf dem i2b2 Datensatz von 2014 erreichte das System einen F1-Score von 88,25% [25].
Folgende Tabelle bietet einen Übersicht über die besten Leistungen aller behandelten Systeme (nach F1-Score absteigend sortiert):

Modell	**F1-Score**
Liu et al. 2017 (BILSTM)	98,28%
Dernoncourt et al. 2016 (BILSTM)	97,85%
Liu et al. 2015 (CRF+Regeln)	94,64%
Chen et al. 2015 (HMM + Dirichlet)	93,00%
Dias 2016 (NER + Second Pass Detection)	88,25%

Tabelle 8: Übersicht über Ergebnisse verschiedener Arbeiten auf dem I2B2 Datensatz 2014

Die Tabelle bestärkt hierbei den Eindruck, dass Ansätze mit BILSTMs die besten Ergebnisse erreichen. Auch verdeutlicht es, wie viel besser die Leistungen der ML-Systeme im Gegensatz zu klassischen Methoden, wie zum Beispiel Neamatullah et al. sie einsetzen, sind.

3.2.2 Named Entity Recognition

Wie in 2.2.3 beschrieben, besitzt der NER-Bereich viele Parallelen zur Aufgabenstellung der Anonymisierung. Und auch dort werden regelmäßig neue Arbeiten veröffentlicht: Gerade in den letzten Jahren sind durch die erhöhte Verbreitung von neuronalen Netzen viele neue Ansätze vorgestellt worden. Dabei sind in den relevanten Arbeiten vor allem 3 Datensätze von Bedeutung. Nachfolgendend werden die Arbeiten anhand der jeweiligen Datensätze vorgestellt und verglichen.

CoNLL-2003

Der CoNLL Datensatz von 2003 [18] wurde erstmals in ”Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition” von Kim Sang et al. vorgestellt. In seiner Grundform enthält er Texte in Englisch (Texte aus Nachrichten von Reuters) sowie in Deutsch (Zeitungsartikel aus der Frankfurt Rundschau). Im späteren Verlauf wurden Texte aus weiteren Sprachen hinzugefügt, die für diese Arbeit aber keine Relevanz besitzen. Zusätzlich zu den NER-Annotationen (PER(son), LOC(ation), ORG(anisation), OTH(er)) wurden noch Features aus einem POS-Tagger sowie einem Chunker [19] zur Verfügung gestellt. Seitdem wird dieser Datensatz in vielen, auch aktuellen Arbeiten, als grundlegendes Leistungsmaß genutzt [82].
In ”Neural Architectures for Named Entity Recognition” zum Beispiel, stellen Lample et al. zwei verschiedene Ansätze auf CoNLL-2003 vor [45]. Der erste Ansatz setzt auf eine Kombination aus bidirektionalen LSTMs mit CRFs, wobei die CRFs dafür eingesetzt werden, eine Schwäche des reinen BILSTM-Ansatzes auszugleichen. Denn dieser trifft die Vorhersage für jedes Label zwar abhängig von allen Eingangsdaten davor sowie danach, doch nicht abhängig von den Labels, die für die vorhergehenden oder folgenden Daten getroffen werden. Ein CRF hingegen hat auf diese Informationen Zugriff und kann so Fehler in

[17] `https://nlp.stanford.edu/software/CRF-NER.shtml`
[18] https://www.clips.uantwerpen.be/conll2003/ner/
[19] Ein Chunker strukturiert Sätze nach ihrer Syntaktischen Struktur und stellt für jedes Wort die Zugehörigkeit in der Hierarchie zur Verfügung [42]

den Labels entdecken. Ein Beispiel hierfür wäre, wenn die Vorhersage für eine Entität nicht mit einem 'B-' sondern mit einem 'I-' Tag beginnt (vergleiche 2.2.1). Für den zweiten Ansatz führen sie einen neuen, auf Transaktionen basierenden Algorithmus ein. Hierbei übernimmt eine sogenannte Stack-LSTM, vorgestellt von Dyer et al. [28], die Verwaltung der Input Daten. Sie verarbeitet Pakete mithilfe von drei verschiedenen Aktionen (Shirt, Reduce, Out). Unter vergleichbaren Arbeiten erhielten sie in Deutsch das beste, in Englisch das zweitbeste Ergebnis. Dabei erreichte die Kombination aus BILSTMs und CRFs 90,94% beziehungsweise 78,76% F1-Score, ihr Transaktions-Ansatz 90,33% und 75,66%. Interessant ist hierbei, dass sowohl ihr Ansatz als auch der vergleichbarer Arbeiten in Deutsch ≈15% schlechtere F1-Scores als in Englischen Texten liefert [45]. Ähnliche Ergebnisse lassen sich auch in dem Paper von Kim Sang et al. sehen, das die damalige Aufgabe zu dem Datensatz eingeführt hat: Während die besten Systeme ≈88,5% F1-Score im Englischen erreichen, liegen die besten Ergebnisse für Deutsch bei ≈72% [82]. Benikova et al. führen dass vor allem auf die Tatsache zurück, dass die Annotierenden des deutschen Datensatzes keine Muttersprachler waren und der Datensatz so Inkonsistenzen aufweist [6]. Dies war eine der Kernmotivationen für den Deutschen NER-Datensatz GermEval, der weiter unten vorgestellt wird. Des weiteren werden daher für den Vergleich der Systeme in diesem Abschnitt nur die Ergebnisse auf dem Englischen Datensatz herangezogen.
Gillick et al. stellen in "Multilingual Language Processing From Bytes" einen alternativen LSTM-Ansatz vor, der die Eingaben byteweise verarbeitet [32]. Dementsprechend nutzen sie keine der normalerweise genutzten Vorverarbeitungsstechnicken aus dem Bereich des NLP, wie zum Beispiel Tokenization, sondern arbeiten direkt auf dem unveränderten Text. Einmal trainierten sie ihr Modell über alle Sprachen hinweg, ein weiteres mal für jede Sprache getrennt. Das erste Modell erreicht hierbei bessere Ergebnisse: 86,50% im Englischen und 76,22% im Deutschen, gegenüber 84,57% bzw. 72,08% F1-Score für das andere Modell. Auch hier zeigt sich die schlechtere Leistung im Deutschen. Generell liefert der Ansatz von Lample et al. bessere Ergebnisse, doch das hier von Gillick et al. vorgestellte Modell ist kompakter und flexibler (da es auf kein Preprocessing baut) [32].
Einen anderen Ansatz stellten Chiu et al. 2015 vor: Sie kombinieren BILSTMs auf Wortebene (wie sie in den obigen Arbeiten häufig verwendet werden) mit CNNs auf Zeichenebene. Dieser Ansatz basiert auf der Arbeit von Santos et al. [71], welche mittels CNNs auf Zeichenebene sehr gute Ergebnisse im Spanischen sowie Portugiesischem NER erreichten. Dabei werden die Buchstaben über Embedding-Technologien, welche auf Zeichenebene eingesetzt werden, an das CNN übergeben. Dieses CNN wird genutzt um Features zu generieren, welche dann als zusätzliche Eingabe für das BILSTM verwendet werden. Auf diese Weise ist es möglich, Strukturen auf Buchstabenebene, welche Wörter einer bestimmten Klasse gemeinsam haben, zu detektieren. Dies können zum Beispiel häufige Endungen von deutschen Städten sein ('-hausen', '-stadt', ...). Ähnliche Ansätze, welche statt CNNs BILSTMs für die Generierung solcher, auf Zeichen basierter Features verwendeten, zeigen laut den Autoren keine besseren Leistungen, benötigen aber mehr Rechenkapazitäten [71]. Unter der Verwendung von Word Embeddings sowie Nachschlagetabellen erreichten sie auf dem Englischen CoNLL-Datensatz einen F1-Score von 91,62% - besser, als alle anderen, oben genannten Arbeiten [17]. Ein auf diesem Paper aufbauendes System wird auch im Rahmen der Evaluierung dieser Arbeit eingesetzt und wird unten im Rahmen des GermEval Datensatzes näher erläutert.
Auf einen ähnlichen Ansatz setzen Ma et al. in "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF", wobei sie zusätzlich zu der Kombination aus BILSTM und CNN noch ein CRF einsetzen, wie bei Lample et al. beschrieben. So erreichen sie auf dem CoNLL-Datensatz einen F1-Score von 91,21%, welcher also um 0.4% schlechter als bei Chiu et al. ist. Doch Chiu et al. trainierten ihr finales Modell nicht nur auf dem Trainings, sondern auch auf dem Entwicklungsset - Ma et al. taten dies nicht. Des weiteren ist der Vorteil des CRFs im Vergleich zu ihren anderen Modellen zu erkennen: Ohne CRF (BILSTM+CNN) erreichte das Modell fast 2% weniger F1-Score (89.36%), ohne CNN nur 87.00% [54]. Da die Dateien für dieses System öffentlich verfügbar sind und mit die besten Leistungen zeigt, wird es auch im Rahmen der Experimente betrachtet.
Eine Übersicht über alle Ergebnisse der erwähnten Arbeiten bietet Tabelle 9.

Modell	F1-Score
Chiu et al. 2015 (BILSTM+CNN)	91,62%
Ma et al. 2016 (BILSTM+CNN+CRF)	91,21%
Lample et al. 2016 (BILSTM + CRF)	90,94%
Lample et al. 2016 (Transaktionen-LSTM)	90,33%
Ma et al. 2016 (BILSTM+CNN)	89,36%
Ma et al. 2016 (BILSTM)	87,00%
Gillick et al. 2015 (Byte-LSTM)	86,50%

Tabelle 9: Übersicht über Ergebnisse verschiedener Arbeiten auf dem CoNLL Datensatz 2003

Aus ihr ist noch einmal ersichtlich, dass Ansätze mit BILSTMs die besten Ergebnisse leisten. Zusätzliche Leistungen können durch weitere Komponenten wie CNNs oder CRFs erreicht werden.

WNUT16

Im Rahmen des 2016 zum zweiten mal gehaltenen W-NUT Workshops [20] wurde eine NER-Aufgabe gestellt, welche sich mit der Detektion in Twitter Tweets beschäftigt - 10 Teams nahmen daran teil. Strauss et al. geben in "Results of the WNUT16 Named Entity Recognition Shared Task" eine Übersicht über alle Einreichungen sowie deren Ergebnisse [78]. Jeder Tweet im Datensatz ist Tokenweise annotiert und zu erkennende Entitäten mit einer von 10 Klassen (z.B. 'Person', 'Unternehmen', 'Ort') gelabelt. Das beste Systeme erreicht im Durchschnitt über 4 Szenarien einen F1-Score von ≈60%. Unter den besten 4 Einreichungen befinden sich zwei Bidirektionale LSTMs (59,63% bzw. 50,26% F1-Score), welche keine zusätzlichen Features verwenden. Das dritte System nutzt Reinforcement-Learning sowie eine breite Auswahl an Features und erreicht so einen F1-Score von 53,13%. Die viertbeste Einreichung ist ein CRF-System, welches ebenso ein großes Set an Features benutzt und einen F1-Score von 47,26% erreicht [78].
Im Vorjahr wurde auch im Rahmen des Workshops eine sehr ähnliche Aufgabe gestellt - die meisten Systeme setzten dabei auf CRFs und erreichten in vergleichbaren Szenarien um bis zu 4% höhere F1-Scores [3]. Im Gegensatz zu 2016, wo ≈1500 Tweets zum Training zur Verfügung standen, konnten die Systeme 2015 mit ≈1800 Tweets trainieren. Im Allgemeinen erreichen die Systeme niedrigere Ergebnisse als ähnliche Ansätze auf dem CoNLL-Datensatz. Dies kann zum einen darauf zurück geführt werden, dass in diesem Falle 10 statt nur 4 Klassen unterschieden werden. Zum anderen lässt dies aber auch darauf schließen, dass unreguläre Daten (vergleiche Sektion 2.3.3), wie Tweets es sind, deutlich schwerer für ML-Systeme zu erfassen sind als reguläre Daten [3].

GermEval 2014

In ihrem Paper "GermEval 2014 Named Entity Recognition Shared Task: Companion Paper" stellen Benikova et al. die Systeme vor, die im Rahmen des KONVENS-Workshops [21] für die sogenannte "GermEval 2014"-Aufgabenstellung eingereicht wurden. Die Aufgabe befasst sich mit NER von 4 verschiedenen Klassen (Ort, Person, Organisation, Sonstiges) in deutschen Wikipedia-Texten. Eine Besonderheit ist hierbei die Annotation von Teilentitäten: Taucht zum Beispiel ein Ort als Teil eines, als 'Sonstige' gelabeltes Sprackkonstrukt auf, wird er mit einer entsprechenden, gesonderten Annotation versehen. Die beiden besten eingereichten Systeme, gemessen an der F1-Score, setzen sich mit ca. 3,5% relativ deutlich vom Mittelfeld ab. Das erste System kombiniert LCRFs mit Statistischen Methoden als auch mit Wortclustern und Query-basierten Features, zum Beispiel für typische Endungen von deutschen Städten wie '-hausen' oder '-stadt'. Sie erreichen so im Schnitt über alle Messmethoden ≈ 78% [36]. Des Weiteren zeichnet sich ihr System durch einen vergleichsweise hohen Recall von ≈ 76,50% aus (≈ 3% mehr als das folgende System). Zweiteres System setzte auf tiefe neuronale Netze in Kombination mit Nachschlagetabellen und erzielte so einen durchschnittlichen F1-Score von ≈ 76,5% [68] [21]. Es zeichnet sich dabei durch eine

[20] Workshop on Noisy User-Generated Text
[21] https://www.oeaw.ac.at/ac/konvens2018/

besonders hohe Precision von ≈ 80% aus [37]. Im Allgemeinen erzielten Systeme, die nur auf Regeln sowie auf Nachschlagetabellen setzten, im Vergleich zu ML-Systemen schlechtere Ergebnisse. Über alle Systeme zeichnete sich ab, dass die Kategorien 'Person' sowie 'Ort' sehr gut erkannt werden. Organisation hingegen werden schlechter erkannt, in der Kategorie 'Sonstiges' wurden mit Abstand die schlechtesten Werte gemessen. Letzteres lässt sich dadurch begründen, dass die Kategorie 'Sonstiges' die mit Abstand Diverseste und somit auch für die Systeme Komplexeste darstellt, während sie in den Trainingsdaten vergleichsweise selten vorkommt. Dass die Ergebnisse insgesamt, zum Beispiel im Vergleich zu ähnlichen Ansätzen auf dem CoNLL-Datensatz (welcher auf einer ähnlichen Art von Texten basiert) deutlich schlechter ausfallen (13% schlechtere F1-Score), lässt darauf schließen, dass NER in Deutsch für ML-Systeme schwerere als im Englischen ist.
Basierend auf selbigen Datensatz veröffentlichten Benikova et al. 2015 ein Tool zur NER, GermaNER genannt. Es ist dabei als Pipeline aufgebaut: Der erste Schritt unterteilt die Eingabedaten in Tokens, der zweite annotiert (intern) die Daten mit diversen Features, welche teilweise Wissen aus externen Quellen einsetzen (zum Beispiel POS-Tagging) [22]. Anschließend werden die annotierten Daten in ein LCRF gegeben. Mit diesem Ansatz erreichten sie einen F1-Score von 76%, wobei sie das Konzept der Teilentitäten außen vor ließen [7]. Da die Dateien für dieses System öffentlich verfügbar sind und es gute Leistungen zeigt, wird es im Rahmen der Experimente dieser Arbeit näher untersucht.
Ein weiteres System, welches in den Experimenten untersucht wird, ist 'German-NER', welches von dem Github-Nutzer isohrab [23] auf Basis der Arbeiten von Chiu et al. sowie von Santos et al. entwickelt wurde (siehe 'CoNLL-2003'). Es setzt dementsprechend auch eine Kombination aus einem BILSTM sowie einem CNN ein und erreicht so auf dem GermEval Datensatz einen F1-Score von 68,73% [40]. Dass das System hierbei einen um 10% schlechteren F1-Score als vergleichbare Ansätze aus dem GermEval-Paper erreicht ist inkonsistent mit den sehr guten Leistungen von vergleichbaren Systemen im Englischen NER. Die Begründung scheint hierbei aber nicht in der Sprache zu liegen, da es im Rahmen der Experimente in dieser Arbeit sehr gute Ergebnisse liefert (siehe Sektion 4.4). Eventuelle Unterschiede in der Messmethodik sind nicht auszuschließen (zum Beispiel der Einsatz von Micro- gegenüber Macro-F1), doch es liegen diesbezüglich keine ausreichenden Informationen vor [40].

3.3 Zusammenfassung

In dieser Sektion wurden relevante Arbeiten aus den Bereichen der Anonymisierung sowie der NER vorgestellt. Neben ML-Ansätzen wurden auch klassische Methoden vorgestellt, darunter das KSystem, das Vergleichssystem aus der Industrie. Während für KSystem keine Vergleichswerte aus anderen Arbeiten vorliegen, liegen durch die Arbeit von Neamatullah et al. mit einem F1-Score von 72,42% Vergleichsergebnisse für klassische Methoden im medizinischen Bereich vor.
Es hat sich herausgestellt, dass BILSTMs in der Anonymisierung von medizinischen Daten die besten Ergebnisse liefern, in der Englischen NER (CoNLL) waren Kombinationen aus BILSTMs mit CNNs am erfolgreichsten. Bei der NER auf Tweets, welche von dem Aufbau der Daten (Web-Daten, erhöhte Frequenz von Rechtschreibfehlern, keine klare Struktur) nahe an dem des Chat-Korpus liegen, haben sich BILSTMs als besonders erfolgreich erwiesen - aber dies hat durch die deutlich niedrigeren F1-Scores auch gezeigt, dass die Erkennung von Named Entities auf solchen Daten deutlich erschwert ist. GermEval 2014 hat des weiteren gezeigt, dass NER auf Deutsch bedeutend schwerer für ML-Systeme als im Englischen ist, wobei hier sowohl CRFs als auch NN gut abgeschnitten haben.
Insgesamt hat sich aber gezeigt, dass in der Anonymisierung, abseits von medizinischen Daten, wenig Arbeit geschieht. Auch in der NER wird zum einen die deutsche Sprache seltener behandelt, zum anderen sind häufig reguläre Daten, wie zum Beispiel Zeitungsartikel im CoNLL-Datensatz, Subjekt der Analysen. Daher ist es das Ziel dieses Arbeit, einen Teil der vorgestellten Systeme auf diese Lücken anzuwenden: Eine Anonymisierung von nicht-medizinischen, unregulären Texten der Deutschen Sprache.

[22] Eine Übersicht über alle Features ist unter `https://github.com/tudarmstadt-lt/GermaNER/blob/master/germaner/src/main/java/de/tu/darmstadt/lt/ner/doc/Features.md` gegeben
[23] `https://github.com/isohrab/German-NER`

4 Experimenteller Aufbau

Die Experimente dieser Arbeit beschäftigen sich damit, einige der in der letzten Sektion vorgestellten Systeme aus der NER auf den Aufgabenbereich der Anonymisierung anzuwenden und sie dabei mit dem System aus der Industrie (KSystem) zu vergleichen. Für den Aufgabenbereich der Anonymisierung stehen keine deutschen Korpera frei zur Verfügung. Daher war es notwendig, Daten zu generieren, um die Ansätze trainieren und testen zu können. Die Erstellung der beiden Datensätze wird in den ersten Abschnitten behandelt. Dabei handelt es sich zum einen um einen Chatkorpus (Dortmund Chat Korpus), zum anderen um einen Datensatz, welcher auf der E-Mail Kommunikation im Bereich 'Kundensupport' basiert.. Anschließend werden die verschiedenen Systeme auf unterschiedlichen Kombinationen der Daten trainiert und ihre Leistung erst allgemein, dann anhand verschiedener Szenarien tiefer gehend analysiert.

4.1 Dortmunder Chat Korpus

Der Dortmunder Chat Korpus umfasst eine Sammlung von 140.240 Chat Nachrichten in 478 verschiedenen Chatverläufen mit insgesamt über 1 Millionen Tokens. Die Sprache ist hierbei fast ausschließlich Deutsch, Ausnahmen bilden hier nur System-Nachrichten in einem Teil der Chatverläufe (zum Beispiel: 'Michaela joined the Room'). Die Nachrichten stammen hierbei aus vier verschiedenen Bereichen: Freizeit, (Studien-)Beratung, Lehr-/Lernkontext (z.B. aus Seminaren) sowie Medien (z.B. Interviews mit Politikern). Die Bereiche teilen sich dabei wie folgt auf [4] [5]:

Teilkorpus	**Mitschnitte**	**Nutzerbeiträge**	**Tokens**
Freizeit	90	88.262	517.828
Lehr-/Lernkontexte	47	12.922	88.833
Beratung	242	21.340	219.345
Medien	99	17.716	237.767
Gesamt	478	140.240	1.063.773

Tabelle 10: Übersicht über die Bereiche des Dortmunder Chatkorpusses

Bei dem für diese Arbeit vorliegenden Korpus handelt es sich um eine anonymisierte Version, in der alle schützenswerten Daten durch IDs ersetzt wurden. Die Durchführung dieser Anonymisierung wurde von Lüngen et al. in ihrem Paper 'Anonymisation of the Dortmund Chat Corpus 2.1' vorgestellt [53]. Diese wurde, basierend auf der Expertise einer Rechtsberatung, manuell von vier Studenten durchgeführt, welche die Entitäten mit IDs aus 13 verschiedenen Kategorien (dazu mehr in 4.1.1) ersetzen. Es handelt sich dabei also um eine Pseudonymisierung, welche durch die fehlenden Dokumente mit den ursprünglichen Werten zu einer Anonymisierung wurde.
Um die Übereinstimmung zwischen den annotierenden Personen zu messen, bearbeiteten alle vier Personen eine bestimmte Datei mit 126 zu ersetzenden Entitäten. Abgesehen von ersterer Person, welche Anweisungen falsch umsetzte (dies aber vor der Annotation des richtigen Korpus änderte), erreichten sie somit eine Übereinstimmung (nach Fleiss' Kappa [24]) von 0,827. Dies kann, basierend auf der Arbeit von Landis und Koch, als "almost perfect agreement" [47] gesehen werden. Zu beachten ist weiterhin, dass bei der Einteilung der Entitäten zwar eine Kategorisierung ähnlich wie bei NER vorgenommen wird, diese aber trotzdem im Rahmen einer Anonymisierung annotiert wurden. Es wurden zum Beispiel weder Namen von Politikern, noch Telefonnummern anonymisiert, welche keine Rückschlüsse auf Teilnehmer zulassen. Auch enthält es Kategorien wie zum Beispiel 'URL', welche normalerweise nicht von NER abgedeckt werden [53]. So zum Beispiel in folgendem Chatverlauf über Radrennen, wo Sven Montgomery nicht als Personenname annotiert ist, da es sich um eine Berühmtheit handelt, über die keine personenbezogenen Daten bekannt gegeben werden:

[24] Fleiss' Kappa ist ein statistisches Maß um die Übereinstimmung zwischen mehreren Beurteilern (hier: Annotierer) [69]

Person A: wer gewinnt diese tour? Person B: sven montgomery

In der NER würde der Name mit einem entsprechenden Label versehen werden.
Wie bereits häufig erwähnt fällt der Korpus in die Kategorien der unregulären Daten: Die Chatverläufe folgen keiner fester Struktur, wie es zum Beispiel bei Wikipedia-oder Zeitungsartikeln der Fall ist (welche häufig Subjekt zum Beispiel in der NER sind, wie in Sektion 3 dargestellt). Des weiteren treten gehäuft Rechtschreibfehler sowie starke Dialekte auf, wie dieses (Extrem-)Beispiel aus dem Korpus zeigt:

ich nimä mal ah dasäs das öppädiä mal git wemer sovil zämä isch, und grad so vor uftritt oder so isches ja au nöd so guät . . .

Die für diese Arbeit vorliegende Version 2.2 des Korpus ist die zu diesem Zeit aktuellste, verfügbare Version, unterscheidet sich aber nur geringfügig von der Version 2.1, welche in dem Paper behandelt wird.

4.1.1 Vervollständigung

Da der Korpus nur in einer anonymisierten Form zugänglich ist, bedurfte es einer Vervollständigung, um ihn zum Training sowie zum Testen der Systeme nutzen zu können. Dabei war es dank der hohen Übereinstimmung zwischen den anonymisierenden Studenten möglich, die Kategorien der dabei ersetzten Entitäten als Labels für den vervollständigten Datensatz zu nutzen. Der Ablauf dieser Vervollständigung wird in diesem Abschnitt näher beschrieben. Um die Qualität der Vervollständigung zu verbessern, wurde eine ältere Version des Korpus [25] hinzugezogen, welche manche der im aktuellen Korpus vorhandenen Chatverläufe in unanonymisierter Form beinhaltet. Auf die genaue Verwendung wird an den jeweiligen Stellen noch einmal eingegangen.
Die Autoren nahmen eine grundlegende Unterteilung in 13 Kategorien vor, welche in Tabelle 11 gegeben ist. Zusätzlich wurde zu den Kategorien Personname sowie Nickname das Geschlecht, falls bekannt, zur Verfügung gestellt. Zu beachten ist dabei, dass sich die Anzahl der Vorkommen auf den Datensatz nach der Vervollständigung beziehen, welche im weiteren Verlauf dieser Sektion behandelt wird. Daher weichen diese Zahlen von den Frequenzen ab, wie sie in der Arbeit von Lüngen et al. berichtet werden.
Zu beachten ist weiterhin, dass die Dichte an Tokens, welche anonymisiert werden müssen, mit 4% relativ gering ist (Dementsprechend gehören zu 96% der Tokens der 'O'-Tag). Im Allgemeinen weist der Korpus ein besonders hohes Vorkommen von 'NICK' auf - dies ist begründet durch dessen häufiges Vorkommen in Systemnachrichten (zum Beispiel: 'Manu12 betritt den Raum'), Anreden (zum Beispiel: '@Manu12' oder 'an Manu12:') sowie im allgemeinen Gebrauch ('Der Martin meinte ja, dass die Bescheide schon im August raus gehen'). Gerade erstere beide Fälle zeichnen sich durch eine sehr reguläre Struktur aus. Sie sollten für die Systeme dadurch leichter zu erkennen sein. Relativ häufig tauchen auch Raumnamen ('ROOM') sowie Internetadressen ('URL') auf. Besonders erstere tauchen fast ausschließlich in Systemnachrichten wie 'Manu12 betritt den Raum RUB-Beratung' auf. Diese reguläre Struktur sollte dazu führen, dass Raumnamen leichter zu erkennen sind. Besonders geringe Vorkommen hingegen weisen 'EMAIL' sowie 'CITATION' auf. Während erstere durch ihr klares Muster trotz der geringen Menge an Vorkommen eine Chance haben, gut erkannt zu werden, gestaltet sich dies bei Zitaten schwerer. Sie werden zwar alle von Anführungszeichen umfasst, doch dies trifft auch auf viele andere Stellen im Datensatz zu, welche nicht entsprechend annotiert sind. Daher ist es nicht zu erwarten, dass sie von den Systemen gut erkannt werden.

[25] `http://www.chatkorpus.tu-dortmund.de/korpora.html`

Kürzel	Beschreibung	Beispiele	Vorkommen
PER	Personname: Ein Erst-, Zweit-oder Vollname einer Person	”Peter Bauer”, ”Mathilde Müller”	848
NICK	Nickname: Nutzername, welche von dem Nutzer frei gewählt werden konnte	”Lela”, ”Martin83!”, ”Captain Marvel”, ”Klaus Organisator”	47.522
ORG	ORGANISATIONNAME: Firma (z.B. Arbeitgeber eines Teilnehmers), Sportverein, Akademisches Organ etc.	”TU Darmstadt”, ”Thyssen Krupp”, ”ASTA-Darmstadt”	606
GPE	GEOPOLITICALENTITTYNAME: Ein Ort, dessen Grenzen offiziell definiert sind, wie zum Beispiel bei Städten, Stadtteilen oder Ländern	”Darmstadt”, ”Hessen”, ”NRW”	1.300
LOC	LOCATIONNAME: Ein Ort oder Gebiet, welcher nicht in die Kategorie GPE fällt, wie zum Beispiel Straßen, Gebirge oder Flüsse	”A5”, ”Herrngarten”, ”Nordschleife”	96
GEO_DE	GEODERIVATIONNAME: Substantiv oder Adjektiv, welches morphologisch von einem Namen (meistens GPE/LOC) abgeleitet ist und eine Assoziation oder Qualität ausdrückt (Adjektive) oder eine Gruppe (z.B. Einwohner einer Stadt) beschreibt (Substantiv)	”Darmstädter”, ”Rednecks”	172
OTH	OTHERNAME: Restkategorie für diverse sensible Begriffe und Referenzen, welche keiner anderen Kategorie zugewiesen werden können	”Unicum” (Zeitschrift), ”GBCF 04/711”	335
ROOM	ROOMNAME: Name des Chatraums	”Lounge”, ”Willkommen”	3.254
URL	WWWURL: Internetadressen	”http://www.ke.tu-darmstadt.de/”	339
EMAIL	EMAIL: E-Mail Adressen	”info@ke.tu-darmstadt.de”	50
NUMBER	NUMBER: Jedwegiger Code oder Nummer, welche mit einer Person in Verbindung gebracht werden können wie Telefonnummern, Alter, Hausnummern, Postleitzahlen, Passwörter, Passnummern, Daten, IP-Adressen. . .	”06151 1621811”, ”64295”, ”192.214.67.12”	166
IMPLICIT	IMPLICIT: Implizite Referenzen sowie Teile von Informationen, welche etwas über Personen preis geben, wie zum Beispiel ihre Arbeit oder Hobbys	”IT-Administrator”, ”Fußballspieler”	122
CITATION	CITATION: Ein Zitat, zum Beispiel aus einem Lied, was genutzt werden kann um eine Person zu identifizieren	”Zeit ist Geld und Geld ist Zeit, doch beides ist heute eher eine Seltenheit”	4

Tabelle 11: Übersicht über die Anonymisierungs-Kategorien des Dortmunder Chatkorpusses [53]

Um die Vervollständigung durchzuführen, wurde ein System in Python entwickelt. Dieses nimmt für die am häufigst vorkommenden Kategorien eine automatische Einsetzung vor und transformiert die Daten in eine geeignete Form, um die Ersetzung der restlichen Entitäten händisch vornehmen zu können. Dabei

wurde eine Analyse der größten Gruppen (NICK, ROOM, OTH, URL, PER, ORG, GPE) vorgenommen um festzustellen, für welche sich Entitäten mit einem angemessenem Aufwand automatisch generieren lassen. Hierbei stellte sich heraus, dass 'Othername' aufgrund des Aufbaus der Kategorie ('Restkategorie') sehr divers und stark abhängig vom Kontext ist. Daher wurde diese Kategorie von der automatischen Generierung ausgeschlossen, um eine hohe Qualität der Ersetzungen zu erreichen. Ähnliches gilt für URLs, welche bewusst sehr divers (mit einer variablen Menge an Unterseiten) und dem Kontext angepasst gewählt werden sollten und sich in ihrer Gesamtmenge in einem Bereich bewegten, der mit einer manuellen Ersetzung zu bewältigen ist. Letztendlich wurde auch 'Organisationname' von der automatischen Generierung ausgeschlossen, da sich auch hier die Möglichkeiten der Entitäten (Firma, Akademische Organisation etc.) stark unterscheiden und bestmöglich dem Kontext angepasst sein sollten. Die restlichen Kategorien werden manuell vervollständigt, da sich aufgrund deren geringer Gesamtmenge das Hinzufügen eines Generators in der automatischen Vervollständigung nicht lohnen würde, vor allem, da man davon ausgehen kann, dass eine händische Ersetzung meist mit einer höheren Qualität eingeht. Somit ergibt sich die in Tabelle 12 dargestellte Einteilung der Kategorien für eine automatische, beziehungsweise händische (manuelle) Vervollständigung.

Kürzel	**Vervollständigung: Automatisch (A) / Manuell (M)**
PER NICK GPE ROOM	A
ORG LOC GEO_DE OTH URL EMAIL NUMBER IMPLICIT CITATION	M

Tabelle 12: Übersicht über die Aufteilung der Kategorien in Automatische und Manuelle Ersetzung

Automatische Vervollständigung

Die automatische Vervollständigung setzt auf eine Mischung aus der Python-Bibliothek Beatiful Soup [26] sowie regulärer Ausdrücke (3.1), um alle anonymisierten Entitäten in den Daten des Korpus zu erkennen. Anschließend wird, basierend auf dem Typ einer jeden Entität, zufällig eine Ersetzung generiert und alle Vorkommen dieser Entität damit ersetzt. Entitäten, welche zur manuellen Ersetzung vorgesehen sind, werden mit einer eindeutigen Identifikation versehen (mehr dazu unter 'Manuelle Vervollständigung'). Als Grundlage für die Ersetzung dient dabei ein Satz von Entitäten, welcher in Tabelle 13 dargestellt ist. Einige von ihnen werden dabei wiederum aus anderen generierten Entitäten zusammen gesetzt. Während die konkrete Entität (z.B. ein bestimmter Name) generell zufällig ausgewählt wird, werden diese meistens durch ein Maß (zum Beispiel ihre Häufigkeit) gewichtet, um einen möglichst repräsentativen Datensatz zu gewährleisten.

[26] Beatiful Soup, eine Python-Bibliothek um XML-ähnliche Daten einzulesen und zu bearbeiten. Mehr unter `https://www.crummy.com/software/BeautifulSoup/`.

Entität	Gewichtung	Beispiel	Quelle
Vorname	Popularität (in Deutschland)	Peter	Datenbank mit 40.000 Namen, inklusive Angabe des Geschlechts - Quelle: https://www.heise.de/ct/ftp/07/17/182/
Nachname	Popularität (in Deutschland)	Müller	Datenbank mit den 1.000 häufigsten deutschen Nachnamen - Quelle: http://wiki-de.genealogy.net/Die_1000_h%C3%A4ufigsten_Familiennamen_in_Deutschland
Voller Name	Gewichtung der Bestandteile	Peter Müller	Eine Kombination aus Vor-sowie Nachname
Nickname	Gewichtung der Bestandteile	Karolina29!	Eine zufällige Kombination aus Vor-/Nachnamen, Zahlen und Satzzeichen (einzelne Namen beinhalten nicht unbedingt jeden Bestandteil)
Stadt	Einwohnerzahl	Darmstadt	Basierend auf einer Liste des Zensus (Stand 2014) - https://gist.github.com/embayer/772c442419999fa52ca1
Bundesland	Einwohnerzahl	Hessen	Basiert auf der Datenbank der Städte
Raumname	-	Lounge	Manuell angelegte Liste, basierend auf den unanonymisierten Chatverläufen

Tabelle 13: Übersicht über die verschiedenen Entitäten, die zur Ersetzung verwendet werden

Zu beachten ist hierbei, dass bei der Generierung von Namen, welche Vornamen beinhalten (zum Beispiel Nickname oder Volle Namen) das Geschlecht, falls bekannt, berücksichtigt wird. Tabelle 14 weist eine Übersicht über den Einsatz die Entitäten in der Ersetzung auf.

Kürzel	Ersetzung	Wahrscheinlichkeiten
PER	Voller Name	100%
NICK	Vorname Nickname Nachname	73% 20% 7%
GPE	Stadt Bundesland	90% 10%
ROOM	Raumname	100%

Tabelle 14: Übersicht über die verschiedenen Ersetzungen für die jeweiligen Entitäten

Die Wahrscheinlichkeiten, nach denen die Ersetzungen vorgenommen werden, orientieren sich an einer Stichprobe aus 4 Dokumenten (im Korpus als 1202001-1202004 gekennzeichnet [27]) aus den unanonymisierten Chatverläufen. Selbiges gilt für die konkrete Zusammensetzung von einzelnen Entitäten, wie zum Beispiel Nicknamen. Eine beispielhafte Tabelle dieser Erhebung für NICK ist in Tabelle 15 gegeben. Rolle bezeichnet hierbei Ausprägungen wie zum Beispiel 'Sekreteriat' - aufgrund der geringen Häufigkeit wurden diese aber nicht in der Vervollständigung berücksichtigt.

[27] Diese Stichprobe wurde gewählt, da sie aus den zur Verfügung stehenden Dokumenten die repräsentativste und regulärsten Strukturen aufgewiesen haben

Ausprägung	1202001	1202002	1202003	1202004	Gesamt
Vorname	13	25	15	14	67
Nachname	2	1	1	2	6
Nickname	4	3	2	5	14
Rollenname	1	1	0	1	3

Tabelle 15: Anzahl der Vorkommen von unterschiedlichen Ausprägungen der Entität Nickname in verschiedenen Dokumenten

Manuelle Vervollständigung

Die manuelle Vervollständigung wurde im Anschluss an die automatische Vervollständigung vorgenommen. Dadurch bestand die Möglichkeit, manuell eingesetzte Entitäten an automatisch generierte anzupassen, sowie kleinere Fehler zu korrigieren, die im Rahmen der automatischen Vervollständigung aufgetreten sind.
Eine besondere Schwierigkeit stellten manche Entitäten der Klasse 'OTHERNAME' dar. Bei einigen von ihnen war es nicht möglich, durch den Kontext auf die konkrete Art der möglichen Ersetzung zu schließen. Erschwert wurde dies durch die Tatsache, dass die Ausprägungen von OTHERNAME im Originalkorpus sehr divers sind. Ein Beispiel für solch einen Fall ist die folgende Nachricht:

07:17 Michaela (1_141_OTHERNAME_4) Quit (Ping timeout)

Systemnachrichten dieser Form treten nur in einem kleinen Teil der Chats auf. Da Chats mit diesen Nachrichten nicht Teil des unanonymisierten Korpus waren, war es nicht möglich, diese korrekt zu ersetzen. Da sie nur einen Spezialfall innerhalb des Korpus darstellen und der Chat auch ohne sie einen sinnvollen Verlauf nahm, wurden alle Entitäten dieser Form entfernt.
Bei der Auswahl der Ersetzungen wurde sich an den vorhandenen, unanonymisierten Chatverläufen orientiert, um den daraus resultierenden Datensatz so realistisch wie möglich zu halten. Konkret bedeutete dies zum einen, dass besonderer Wert auf die Konsistenz gelegt wurde. Und zwar sowohl mit dem Kontext, in dem die entsprechende Ersetzung statt findet, als auch bei erneutem Vorkommen einer Entität, sodass sie an allen Stellen gleich ersetzt wird. Zum anderen wurde auf eine realistische Diversität geachtet. So wurden zum Beispiel Telefonnummern in mehreren Formaten ersetzt ('06151 1621811', '06151/1621811', '06151-1621811', '6151 162 18 11' et cetera) oder Organisationen, falls passend, auch mal mit ihren Kurzformen erwähnt (zum Beispiel TUD statt TU-Darmstadt). Auch dies hat sich so aus den Originaldaten ergeben. Alle Ersetzungen, welche vorgenommen worden sind, wurden in einer gesonderten Tabelle festgehalten. Auf diese Weise muss bei einem erneuten Durchlauf der automatischen Vervollständigung die manuelle Ersetzung nicht erneut vorgenommen werden.
Nach der Vervollständigung werden die Annotationen noch entsprechend kodiert. Die folgenden zwei Nachrichten zeigen beispielhaft, wie Chatverläufe nach der Vervollständigung aussehen:

ich \O möchte \O in \O Mainz \B-GPE Englisch \O und \O in \O Göttingen \B-GPE, \O wenn \O ' \O s \O klappt \O Bio \O studieren \O , \O muss \O ich \O mich \O dann \O erst \O in \O Mainz \B-GPE einschreiben \O und \O dann \O nach \O Göttingen \B-GPE oder \O ist \O das \O egal \O ? \O

an \O Artur \B-NICK ! \I-NICK : \O Anglistik \O ist \O ein \O 2 \O - \O Fächer \O - \O Bachelor \O , \O Sie \O brauchen \O ein \O 2 \O . \O Fach \O . \O Es \O gehen \O alle \O Faecher \O der \O RUB \B-ORG , \O die \O einen \O 2 \O - \O Faecher \O BA \O anbieten \O , \O da \O gibt \O es \O keine \O Einschraenkungen \O . \O

Ein interessantes Detail ist hierbei bei der Tokenisierung zu sehen: Dadurch, dass der Nickname ein Satzzeichen enthält ('!'), wird er von dem Tokenisierer in 2 Stücke aufgeteilt. Solche Satzzeichen von Satzzeichen, welche in einem normalen Kontext verwendet werden zu unterscheiden, wird eine besondere Herausforderung für die Systeme darstellen.

4.1.2 Aufteilung in Trainings-, Entwicklungs- sowie Testdaten

Nachdem die Vervollständigung abgeschlossen war, musste noch die Aufteilung des Datensatzes in Trainings-, Entwicklungs- sowie Testdaten geschehen (siehe Sektion 2.3.3). Dabei galt es zu beachten, die relative Aufteilung der Klassen zwischen den Datensätzen zu erhalten. Doch auch die Verteilung der verschiedenen Domänen (Tabelle 10) sollte repräsentativ erfolgen, da die Art der Kommunikation zwischen den Domänen deutliche Unterschiede aufweist. Begingt dadurch, dass einige Klassen sehr geballt auftreten (zum Beispiel bei 'CITATION'), galt es diese beiden Gesichtspunkte zu vereinen. Des weiteren ist für die Qualität der Vorhersagen bei den verwendeten Techniken (RNNs sowie CRFs) der Kontext von essentieller Wichtigkeit. Als Resultat daraus konnte der Datensatz nicht in zu kleine Stücke unterteilt werden.

Label	Anzahl 'B-'	Anzahl 'I-'	Gesamt
NICK	33.764	39.86	37.750
PER	594	601	1.195
GPE	916	295	1.211
OTH	260	1	261
GEO_DE	123	0	123
URL	243	0	243
ORG	454	30	484
EMAIL	39	256	295
ROOM	1.643	0	1.643
NUMBER	111	75	186
IMPLICIT	87	1	88
LOC	67	3	70
CITATION	3	8	11
O	1.073.472	-	1.073.472
Gesamt	**1.111.776**	**5.256**	**1.117.032**

(a) Trainingsdatensatz

Label	Anzahl 'B-'	Anzahl 'I-'	Gesamt
NICK	6.750	891	7.641
PER	130	130	260
GPE	183	58	241
OTH	36	2	38
GEO_DE	27	0	27
URL	46	0	46
ORG	81	10	91
EMAIL	5	34	39
ROOM	363	0	363
NUMBER	31	26	57
IMPLICIT	11	0	11
LOC	20	1	21
CITATION	0	0	0
O	214.089	-	214.089
Gesamt	**221.772**	**1.152**	**222.924**

(b) Entwicklungsdatensatz

Label	Anzahl 'B-'	Anzahl 'I-'	Gesamt
NICK	7.008	842	7.850
PER	124	127	251
GPE	201	59	260
OTH	39	4	43
GEO_DE	22	0	22
URL	51	0	51
ORG	71	8	79
EMAIL	6	46	52
ROOM	348	0	348
NUMBER	24	21	45
IMPLICIT	24	4	28
LOC	9	0	9
CITATION	1	7	8
O	213.890	-	213.890
Gesamt	**221.818**	**1.118**	**222.936**

(c) Testdatensatz

Tabelle 16: Überblick über die Frequenzen des Dortmunder Chat Korpus nach der Aufteilung in Datensätze

Als bestmögliche Lösung hat sich die Unterteilung des Korpus in Stücken der Größe von ca. 750 Tokens herausgestellt, wobei die Trennung immer am Ende von Chatnachrichten vorgenommen wurde, um auch an dieser Stelle möglichst viel Kontext zu erhalten. Aus jeweils 7 solcher Stücke wurden 5 für den Trainingsdatensatz und jeweils ein weiteres für Entwicklungs-, sowie Testdaten verwendet. Die Ergebnisse hiervon sind in Tabelle 16 zu sehen. Zu beachten ist, dass der Entwicklungsdatensatz zum Beispiel kein Vorkommen von 'CITATION' besitzt, der Testdatensatz nur ein einzelnes. Dies war, aufgrund der kleinen Größe dieser Klasse (4 Vorkommen im finalen Datensatz), nicht anders zu machen. Denn ein gewisses Vorkommen im Trainingsdatensatz ist notwendig, damit die Systeme die Chance haben, das Konzept dieser Klasse zu erlernen.

4.2 E-Mail Korpus

Da die Daten des oben vorgestellten Dortmunder Chat Korpus wie erwähnt vor allem unregelmäßige Texte darstellen und nur eine Form der Kommunikation (Chats) abdecken, wurde im Rahmen dieser Arbeit ein weiterer Korpus erstellt, welcher E-Mails aus dem Bereich des Kundensupports enthält. Die Erstellung des Korpus beruht dabei auf handgeschriebenen E-Mail Vorlagen, welche automatisch mit passenden Ersetzungen gefüllt werden.

4.2.1 Vorlagen

Die Auswahl der Anwendungsfälle für die Vorlagen wurde in Kooperation mit Experten aus der Industrie getroffen. Hierbei wurde neben einer hohen Praxisrelevanz Wert darauf gelegt, dass alle Anwendungsfälle ausreichend Daten zur Anonymisierung beinhalten. Neben E-Mails aus dem Bereich der Kunden-Firmenkommunikation, welche sich vor allem auch durch eine sehr förmliche Struktur auszeichnen, wurden auch einzelne Anwendungsfälle aus der Firmeninternen Kommunikation ausgewählt, welche eine weniger förmliche Struktur aufweisen. Die Anzahl von Vorlagen dieser Art wurde bewusst gering gehalten, da diese nicht den Schwerpunkt dieses Korpus darstellen sollen.
Im Allgemein existieren für jeden Anwendungsfall ein oder mehrere Vorlagen, welche sich in ihrem Aufbau grundsätzlich untereinander unterscheiden. Von jeder dieser Vorlage wiederum kann es noch weitere Variationen geben, welche nur leichte Änderungen der Vorlage beinhalten. Eine Vollständige Übersicht über alle Anwendungsfälle ist in folgender Tabelle gegeben:

Bereich	**Anwendungsfall**	**Vorlagen**	**Variationen**
Kunde an Firma	Kündigung eines Vertrages	2	2
	Widerruf eines Vertrages	2	2
	Übersendung von persönlichen Daten	1	1
	Adressänderung	3	4
	Änderung von Zahlungsdaten	2	3
	Überprüfung von persönlichen Daten	2	4
	Löschung von Daten nach DSGVO	1	1
	Änderung der Anstellung	1	1
	Änderung einer Abschlagszahlung	2	4
Firmen-intern	Kommunikation von Kundendaten	1	1
	Kommunikation von Zahlungsdaten	1	2
Gesamt		18	25

Tabelle 17: Übersicht über die Anwendungsfälle des E-Mail Korpus

Bei der Formulierung der Vorlagen wurde sich an Beispiel-Texten aus diversen Quellen orientiert, wie zum Beispiel dem Verbraucherschutz. Stellen, an welchen Einsetzungen vorgenommen werden sollen, wurden mit Schlüsseln der Form '{_typ_}' versehen. Eine Übersicht über alle verfügbaren Typen ist in der nächsten Sektion gegeben.

Als Beispiel ist die folgende Vorlage aus 'Kündigung eines Vertrages' gegeben, welche auf Basis eines Musters von Örag [28] erstellt wurde:

> {_formal_salutation_},
>
> *unter Einhaltung der vertraglich vereinbarten Kündigungsfrist kündige ich den {_contract_} zum {_date_}, hilfsweise zum nächstmöglichen Kündigungszeitpunkt. Ich bitte um schriftliche Bestätigung, dass Sie die Kündigung erhalten haben.*
>
> {_formal_regards_}

4.2.2 Vervollständigung

Das System, welche die Vervollständigung des Korpus vornahm, wurde ebenfalls in Python geschrieben und in das für den Dortmunder Korpus verwendetem System integriert. Dadurch konnten sie gemeinsam auf Ressourcen, wie zum Beispiel zur Namensgenerierung, zugreifen. Dementsprechend konnten die Ersetzungen für einige Entitäten, wie Städte oder Namen, direkt aus denen des Dortmunder Korpus gewonnen werden. Für einige andere (wie zum Beispiel Telefon-oder Kreditkartennummern), welche im Dortmund Korpus nicht enthalten waren oder manuell ersetzt worden sind, mussten neue Generatoren definiert werden. Für die Labels hingegen wurden ausschließlich die in Tabelle 11 gegebenen, im Dortmund Korpus verwendete Kategorien, genutzt. Alle zur Nutzung in den Vorlagen verfügbaren Typen sind in Tabelle 18 zusammen gefasst.

Entität	**Ersetzung**	**Label**
full_name	Zufälliger Name, bestehend aus Anrede (Frau/Herr), Vor-sowie Nachname	'PER'
first_name	Zufälliger Vorname	'PER'
formal_salutation	Formelle Begrüßung, zum Beispiel 'Sehr geehrte Damen und Herren' oder 'Sehr geehrte Frau {_full_name_}'	'PER' für eventuell enthaltene Namen, 'O' für den Rest
informal_salutation	Informelle Begrüßung, zum Beispiel 'Hallo {_first_name_}'	'PER' für eventuell enthaltene Namen, 'O' für den Rest
formal_regards	Formelle Verabschiedung, zum Beispiel 'Mit freundlichen Grüßen \n {_full_name_}'	'PER' für enthaltene Namen, 'O' für den Rest
informal_regards	Informelle Verabschiedung, zum Beispiel 'Liebe Grüße \n {_first_name_}'	'PER' für enthaltene Namen, 'O' für den Rest
informal_regards	Informelle Verabschiedung, zum Beispiel 'Liebe Grüße \n {_first_name_}'	'PER' für enthaltene Namen, 'O' für den Rest
name_salutation	Name, welcher in der Begrüßung verwendet wurde	'PER'
name_regards	Name, welcher in der Verabschiedung verwendet wurde	'PER'
id_number	Nummer einer (in gegebenen Grenzen) zufälligen Länge, welche an jeder Stelle eine zufällig ausgewählte Ziffer (0-9) beinhaltet	'NUMBER'

[28] Rechtsschutz-Versicherung Örag, `www.oerag.de`

contract	Verschiedene Variationen, zum Beispiel 'Vertrag (Nummer: {_id_number_})' oder 'Vertrag zugehörig zu der Kundennummer {_id_number_}' (Nummern der Länge 6-12)	'NUMBER' für die Nummern, 'O' sonst
date	Zufälliges Datum der aktuellen Zeit (Jahre 2014-2019)	'NUMBER'
date_birth	Zufälliges Geburtsdatum (Jahre 2014-2019)	'NUMBER'
date_datum	Zufälliges Datum in Kurzform (z.B. 12/19) der aktuellen Zeit (2018-2022), zum Beispiel als Ablaufdatum einer Kreditkarte	'NUMBER'
city	Zufällige Stadt	'GPE'
bank_account	Zufällig generierte Informationen über ein Bankkonto, inklusive Kontonummer (id_number der Länge 8-10) sowie Bankleitzahl (id_number der Länge 8). Dabei wird zufällig aus verschiedenen Formatierungen ausgewählt (z.B: als IBAN, aufgetrennt in Kontonummer und Bankleitzahl sowie verschiedene Trennungen von Zahlenblöcken)	'NUMBER' für enthaltene Nummern, 'O' für den Rest
bank_account_full	Wie bank_account, aber formatiert über mehrere Zeilen	'NUMBER' für enthaltene Nummern, 'O' für den Rest
organisation	Zufällig aus einer gegebeben Liste ausgewählter Firmenname	'ORG'
full_address	Zufällig generierte Adresse, welche mindestens Straße, Hausnummer sowie Ort enthält - zufällig auch noch Postleitzahl sowie Land	'NUMBER', 'LOC' sowie 'GPE'
email_regards	Zufällig generierte E-Mail, welche den Namen aus der Verabschiedung nutzt	'ORG'
telefon_number	Zufällig generierte Telefonnummer, mit Vorwahlen der Länge 3-5 ('0' mitgezählt), Hauptnummern der Länge 4-8 sowie verschiedenen Formatierungen (z.B. 069-543234 oder 069 543234)	'NUMBER'
money_smaller	Zufällig generierte Menge an Geld (in gegebenen Bereich, inklusive €/Euro), wobei der Betrag immer geringer ist als money_bigger	'NUMBER' für den Betrag, 'O' für den Rest
money_bigger	Simultanes Verfahren wie bei money_smaller, wobei der Betrag immer größer ist als money_smaller	'NUMBER' für den Betrag, 'O' für den Rest

Tabelle 18: Übersicht über die im E-Mail Korpus verwendeten Entitäten

Durch die Verwendung von Schlüssel wie 'email_regards' wird die inhaltliche Konsistenz innerhalb jeder E-Mail garantiert.

4.2.3 Resultat

Für jede Variation der Vorlagen wurde die Vervollständigung 9-mal mit verschiedenen Ersetzungen durchgeführt. Auf diese Weise ist es möglich, durch die zufälligen Komponenten in der Vervollständigung eine größere Menge an Daten zu generieren, ohne dass diese sich zu sehr ähneln. Damit ergeben sich die folgenden Frequenzen an Labeln:

Label	Anzahl 'B-'	Anzahl 'I-'	Gesamt
PER	440	584	1.024
GPE	115	27	142
ORG	9	11	20
EMAIL	18	88	106
NUMBER	551	797	1.348
LOC	45	40	85
O	8.636	-	8.861
Gesamt	**9.814**	**1.547**	**11.361**

Tabelle 19: Übersicht über die Frequenzen der Labels im E-Mail Korpus

Hierbei lässt sich gut der Schwerpunkt des Korpus erkennen: Im Vergleich zum Dortmunder Korpus sind die relativen Häufigkeiten von 'PER', 'EMAIL' sowie 'NUMBER' deutlich erhöht. Dabei ist die erhöhte Häufigkeiten von PER auf die vielen Namen in Anreden sowie Verabschiedungen zurückführen, viele 'NUMBER'-Annotationen sind durch Zahlungsdaten (Bankdaten/Kreditkartennummern) sowie Geburtstage begründet. Des weiteren kommen manche Klassen, wie 'IMPLICIT' oder 'CITATION', gar nicht vor, da es keine entsprechenden Entitäten innerhalb der Anwendungsfälle gab. Die Klasse 'ORG' besitzt eine sehr geringe Frequenz, da sie nur in einer Vorlage enthalten war. Die Integration dieser Entität in mehrere Vorlagen war aus realistischer Sicht im Bezug auf die Anwendungsfällen nicht möglich. Die Klasse 'EMAIL' hingegen sticht in sofern heraus, dass sie ein deutlich erhöhtes Vorkommen von 'I'-Tags im Vergleich zu 'B-'-Tags aufweist. Dies ist damit zu begründen, dass durch die Tokenisierung, welche auch an '@' sowie '.' trennt, jede E-Mail in mindestens 5 Tokens aufgeteilt wird. Weiterhin ist zu beachten, dass der Korpus in seiner Gesamtgröße weniger als ein hundertstel der Tokens des Dortmunder Korpus besitzt. Doch während in diesem 96% der Tokens der Klasse 'O' angehören, ist dies im E-Mail Korpus nur bei 78% der Fall. Die Dichte an zu anonymisierenden Entitäten ist im Vergleich dementsprechend stark erhöht. Dies ist darauf zurück zu führen, dass dieser Korpus bewusst für diese Aufgabe gebaut und dementsprechend Anwendungsfälle mit einer hohen Dichte an zu anonymisierenden Entitäten gewählt wurden. Beim Dortmund Korpus hingegen ist das nicht der Fall.
Es folgt ein Beispiel für eine automatische Vervollständigung sowie den Annotationen der oben vorgestellten Vorlage:

> *Sehr\O geehrte\O Damen\O und\O Herren\O,\O*
>
> *unter\O Einhaltung\O der\O vertraglich\O vereinbarten\O Kündigungsfrist\O kündige\O ich\O den\O Vertrag\O zugehörig\O zu\O der\O Kundennummer\O 768006697513\B-NUMBER zum\O 14\B-NUMBER .\I-NUMBER 09\I-NUMBER .\I-NUMBER 2015\I-NUMBER,\O hilfsweise\O zum\O nächstmöglichen\O Kündigungszeitpunkt\O .\O Ich\O bitte\O um\O schriftliche\O Bestätigung\O ,\O dass\O Sie \O die\O Kündigung\O erhalten\O haben\O .\O*
>
> *Mit\O besten\O Grüßen\O Helene\B-PER Dick\I-PER*

4.2.4 Aufteilung in Trainings-, Entwicklungs- sowie Testdaten

Wie in Sektion 2.3.3 beschrieben, ist es für den Einsatz von ML-Modellen notwendig, den Datensatz in Trainings-, Entwicklungs- sowie Testdaten aufzuteilen. Da es sich um einen vergleichsweise kleinen Datensatz handelt, machen Test-und Entwicklungsdatensatz hierbei einen außergewöhnlich hohen Anteil aus. Dies ist notwendig, um trotz der kleinen Größe möglichst repräsentative Ergebnisse erhalten zu können. Des weiteren ist der Trainingsdatensatz ausschließlich dazu gedacht, in Kombination mit dem Dortmund Chat Korpus zum Training eingesetzt zu werden. Daher ist dort eine verringerte Größe nicht so ausschlaggebend. Um die Verteilung gleichmäßig und jeden Datensatz möglichst repräsentativ zu halten, wurde je 2 der 9 Vorlagen für den Entwicklungs- beziehungsweise Testdatensatz verwendet, die restlichen 5 für den Trainingsdatensatz.

Einen genauen Überblick über die Frequenzen der verschiedenen Klassen nach der Aufteilung ist in Tabelle 20 zu sehen.

Label	Anzahl 'B-'	Anzahl 'I-'	Gesamt
PER	242	324	566
GPE	63	16	79
ORG	5	6	11
EMAIL	10	48	58
NUMBER	306	445	751
LOC	25	29	54
O	4.802	-	4.802
Gesamt	**5453**	**868**	**6321**

(a) Trainingsdatensatz

Label	Anzahl 'B-'	Anzahl 'I-'	Gesamt
PER	100	132	232
GPE	27	7	34
ORG	2	3	5
EMAIL	4	22	26
NUMBER	124	174	298
LOC	10	6	16
O	1.918	-	1.918
Gesamt	**2.185**	**344**	**2.529**

(b) Entwicklungsdatensatz

Label	Anzahl 'B-'	Anzahl 'I-'	Gesamt
PER	98	128	226
GPE	25	4	29
ORG	2	2	4
EMAIL	4	18	22
NUMBER	121	178	299
LOC	10	5	15
O	1.916	-	1.916
Gesamt	**2.176**	**335**	**2.511**

(c) Testdatensatz

Tabelle 20: Überblick über die Frequenzen des E-Mail Korpus nach der Aufteilung in Datensätze

4.3 Aufbau

Es werden insgesamt 6 verschiedene Systeme getestet - 5 davon aus dem Bereich der NER sowie das Vergleichssystem KSystem aus der Industrie.

Ma et al. 2016 Das in Sektion 3.2.2 vorgestellte System von Ma et al. wird in drei verschiedenen Ausführungen (als BILSTM, BILSTM_CNN sowie als BILSTM_CNN_CRF) getestet. Es gehört zu den Systemen mit den besten Leistungen in Englischer NER, ist frei verfügbar [29] und deckt mit seinen drei Ausführungen alle Konzepte ab, welche in der NER momentan die besten Leistungen erbringen [54].

German-NER Des weiteren wird das im Rahmen des GermEval vorgestellten 'German-NER'-System (siehe Sektion 3.2.1) eingesetzt. Es dient als zusätzlicher Vergleich gegenüber den Systemen von Ma et al., um das BILSTM_CNN als erfolgversprechenden Ansatz aus der NER in verschiedenen Ausführungen testen zu können. Dadurch kann der Einfluss eines bestimmten Modell-Setups minimiert werden [40]. Zur eindeutigen Identifikation wird das System folgend als 'BILSTM_CNN_2' referenziert.

GermaNER Als Vergleich für ein LCRF wird das im Rahmen des GermEval vorgestellte 'GermaNER' herangezogen (siehe Sektion 3.2.1), da es für die Deutsche Sprache konzipiert wurde. Dies ist im Falle von CRFs relevant, da einige sprachen-spezifische Feature Funktionen existieren [7]. Es wird im folgenden entsprechend als 'LCRF' refenziert.

KSystem Dass in Sektion 3.1 vorgestellte 'KSystem' wird als Vergleich für Klassische Methoden eingesetzt.

[29] Unter `https://github.com/bamtercelboo/pytorch_NER_BiLSTM_CNN_CRF`

Dabei wird jedes der ML-Systeme (also alle außer KSystem) einmal ausschließlich auf dem Dortmund Chat Korpus und einmal auf einer Kombination aus beiden Datensätzen trainiert. Getestet werden sie jeweils aber auf beiden Datensätzen. Zur Unterscheidung erhalten die Systeme, welche auf beiden Datensätzen trainiert wurden, das Suffix '_COMP' für 'complete'. Damit ergibt sich die folgende Aufstellung aller 11 Systeme:

- BILSTM (Ma et al. 2016)
- BILSTM_COMP (Ma et al. 2016)
- BILSTM_CNN (Ma et al. 2016)
- BILSTM_CNN_COMP (Ma et al. 2016)
- BILSTM_CNN_CRF (Ma et al. 2016)
- BILSTM_CNN_CRF_COMP (Ma et al. 2016)
- BILSTM_CNN_2 (German-NER)
- BILSTM_CNN_2_COMP (German-NER)
- LCRF (GermaNER)
- LCRF_COMP (GermaNER)
- KSystem

Von einem Training ausschließlich auf dem E-Mail Korpus wurde aufgrund fehlenden Mehrwertes bewusst abgesehen. Ein exemplarischer Versuch, in dem das BILSTM_CNN_2 ausschließlich auf dem E-Mail Korpus trainiert wurde, führte trotz einer stark erhöhten Anzahl an Epochs zu schlechteren Ergebnissen auf dem E-Mail Korpus als bei BILSTM_CNN_2_COMP, wobei letzteres auch noch in der Lage ist, Klassifikationen auf dem Dortmund Chat Korpus durchzuführen. Dies kann auf die zu geringe Größe des E-Mail Korpus zurück geführt werden. Die Ergebnisse für den exemplarischen Versuch sind unter dem Namen BILSTM_CNN_2_ONLY_EMAIL im Abschnitt B des Anhangs zu finden.
Sowohl die Systeme von Ma et al., als auch das BILSTM_CNN_2, benötigen Word Embeddings, um die Eingaben zu verarbeiten. Dafür wurden die in Sektion 2.2.2 vorgestellten GermanWordEmbeddings verwendet [24].
Als Hyperparameter werden die Werte verwendet, die für das jeweilige System im Bereich der NER in den besten Ergebnissen resultierten. Eine Ausnahme bildet hierbei die Anzahl der 'Epochs', welche für die Systeme von Ma et al. auf 25, für das BILSTM_CNN_2 auf 15 festgesetzt wurden. Hierbei waren vor allem die anderweitig stark erhöhten Trainingszeiten ausschlaggebend, welche bei dem Training von 10 verschiedenen Modellen erheblich ins Gewicht gefallen wären. Die Epochs wurden so gewählt, dass die 4 Modelle eine ähnliche Trainingszeit zur Verfügung hatten und so faire Voraussetzungen herrschten. Des weiteren ergab die Durchführung weiterer Epochs nur noch minimale Verbesserungen auf dem Entwicklungsdatensatz - Ziel dieser Arbeit ist schließlich nicht das Erreichen des besten Ergebnisses, sondern die Erforschung, ob sich diese Modelle auf den Bereich der Anonymisierung übertragen lassen. Aus selbigen Gründen wurde auch auf den Einsatz von Techniken wie Kreuz-Validierung [30] oder Rastersuche [31]

[30] Im Rahmen einer k-fachen Kreuz-Validierung werden der Trainings- sowie Entwicklungsdaten zusammen gefasst. Anschließend werden sie in k gleichgroße Teile aufgeteilt (k ist hierbei frei wählbar). Daraufhin wird das ML-System auf $k-1$ Teilen trainiert und auf einem Teil getestet. Danach werden die Teile durchgetauscht, bis alle Kombinationen einmal verwendet wurden. So kann die Fähigkeit eines Modells zur Generalisierung auf verschiedenen Daten getestet werden, ohne den Trainingssatz zu verkleinern [14].

[31] Eine Rastersuche (Grid-Search im Englischen) ist eine Technik zur Hyperparameter-Optimierung. Es werden alle möglichen Kombinationen von Hyperparametern auf dem Entwicklungsdatensatz getestet. Die Werte, welcher ein Hyperparameter annehmen kann, müssen dafür vordefiniert sein. Das Modell, welches hierbei die besten Ergebnisse erreicht, wird ausgewählt. Häufig wird Rastersuche auch in Kombination mit Kreuz-Validierung eingesetzt [16].

verzichtet. Gerade die Durchführung einer Cross-Validation hätte durch die ungleichmäßige Verteilung der Entitäten im Dortmunder Chat Korpus, wie sie in Sektion 4.1 beschrieben ist, zu weiteren Problemen führen können. Schließlich wurde für jedes System nach der Durchführung das Modell des Epochs gewählt, welches die besten Leistungen auf dem Entwicklungsdatensatz aufweisen konnte.

4.4 Evaluation

In diesem Abschnitt werden die Testergebnisse vorgestellt und analysiert. Dafür wird zuerst das Augenmerk auf die allgemeinen Ergebnisse gelegt, bevor in einer tiefer gehenden Analysen die Vor-, sowie Nachteile der Modelle genauer heraus gearbeitet werden. Dabei werden verschiedene Fragestellungen beantwortet, zum Beispiel, ob die Systeme den Unterschied zwischen einem Personennamen, der anonymisiert werden muss und einem, der nicht anonymisiert werden muss, erkennen können. Zu diesen Zwecken werden dabei diejenigen Metriken verwendet, die besonders dafür geeignet sind, die jeweilige Fragestellung zu beantworten.
Darüber hinaus sind für jede Kombination eines Systems mit einem der beiden Testdatensätze im Anhang ausführliche Metriken gegeben. Diese umfassen das folgende:

Konfusionsmatrix Die Konfusionsmatrix (siehe Sektion 2.3.10) für alle Klassen - die Diagonale, welche richtige Vorhersagen enthält, ist 'fett' gedruckt

Multiklassen-Metriken Darauf folgen 2 Tabellen mit verschiedenen Multiklassen-Metriken (Namentlich: Accuracy, Precision (P), Recall (R), F1-Score sowie Matthews Correlation Coefficient (MCC); Sektion 2.3.10). Wo verschiedene Varianten durch Micro- beziehungsweise Macro-Averaging möglich sind, sind beide Werte angegeben. Dabei wurden diese Metriken einmal über jede Klasse erhoben (erstere Tabelle) und einmal über alle Klassen exklusive der 'O' Klasse (zweitere Tabelle). Dies hat den Grund, dass die 'O'-Klasse den mit Abständen größten Teil der Daten ausmacht, insbesondere im Dortmunder Chat Korpus, ohne dass sie zu anonymisierende Entitäten beinhaltet.

Binäre Metriken Die letzte Tabelle gibt Auskunft über Metriken für die 'binarisierte' Version der Klassifizierung, in welcher die 'O'-Klasse als Klasse '0' und alle anderen Klassen als Klasse '1' behandelt werden. Auf diese Weise kann der Einfluss von Verwechslungen der 'zu anonymisierenden'-Klassen untereinander ausgeschlossen werden. Außerdem kann so, die vor allem aus rechtlicher Sicht interessante Frage, wie viel von allen zu anonymisierenden Daten tatsächlich anonymisiert wurden (unabhängig des bestimmten Typs), geklärt werden. Als Metriken werden hierfür Accuracy, Precision, Recall/True-Positive-Rate (TPR), False-Positive-Rate (FPR), F1-Score sowie MCC zur Verfügung gestellt. TPR sowie FPR werden auch dafür verwendet, die Klassifizierer im ROC-Space anzuordnen (siehe Sektion 2.3.10).

Zu beachten ist, dass im Falle des 'KSystem' teilweise leicht veränderte Klassenfrequenzen vorliegen, insbesondere im Falle der Klasse 'URL'. Dies rührt daher, da die Tokenisierung im Gegensatz zu den anderen Systemen im 'KSystem' intern vorgenommen wurde. Als Folge dessen konnte bei diesem System kein Einfluss auf die Art der Tokenisierung genommen werden, wodurch die Texte unterschiedlich tokenisiert wurden. Dadurch wurde unter anderem das Verhältnis von 'B-' zu 'I-' Labeln verändert.

4.4.1 Gesamtergebnis

Um einen ersten Eindruck über die Gesamtleistung der verschiedenen Systeme zu erhalten, wird die Platzierung der Systeme im ROC-Space herangezogen. Darauf folgend wird ein genauerer Blick auf die weiteren Metriken geworfen. Die Platzierung basiert auf den TPR sowie FPR Werten der Systeme auf der binären Klassifikation und nimmt gerade durch die Vereinfachung auf die binäre Klassifikation einen wichtigen Stellenwert ein. Schließlich ist es für den grundlegenden Erfolg der Anonymisierung vor allem wichtig, wie viele der zu anonymisierenden Wörter gefunden werden. Zweitrangig ist erstmal, ob die vorhergesagte Klasse korrekt ist.

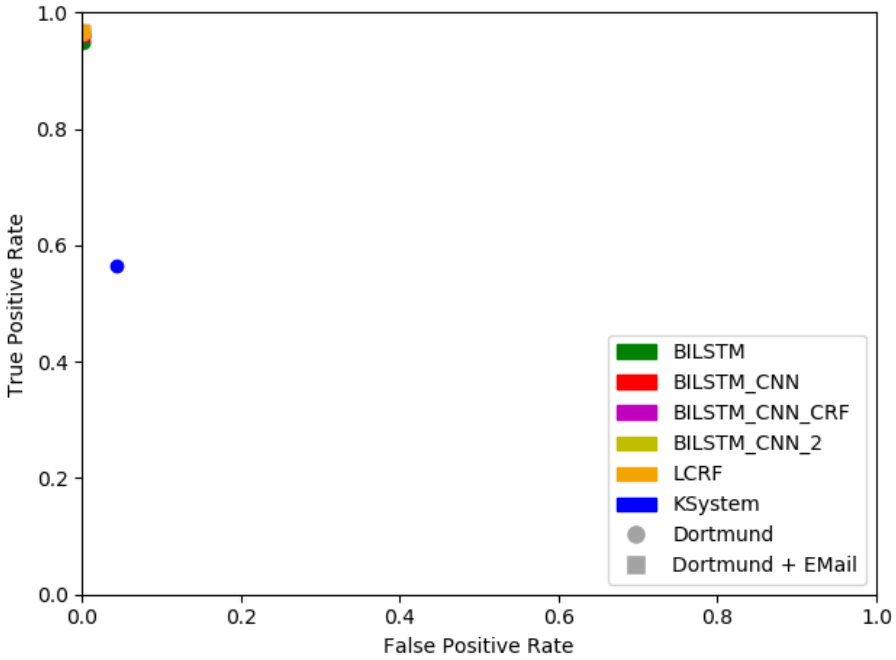


Abbildung 18: Die Leistungen der verschiedenen Systeme auf dem Dortmund Chat Korpus im ROC-Space

Um die Systeme eindeutig identifizieren zu können, erhält jedes System eine eindeutige Farbe. Die Art des Trainingssatzes ist durch die Form der Markierung gegeben: Kreise für Systeme die nur auf dem Dortmund Korpus trainiert wurden, Quadrate für Systeme, die auf beiden Korpera (Suffix _COMP) trainiert wurden. Da das KSystem kein Training erhalten hat, wird es nur durch Kreise repräsentiert.

Dortmund Chat Korpus

Der ROC-Space für den Test auf dem Dortmund Chat Korpus ist in Abbildung 18 dargestellt. Es fällt ins Auge, dass die meisten Systeme in der oberen, linken Ecke konzentriert sind - sie besitzen somit alle einen hohen Recall und eine sehr niedrige FPR. Ersteres verrät also auch, dass die Systeme sehr wenige falsch-negative aufweist - dies ist im Falle der Anonymisierung besonders wichtig, da ein fälschlicherweise 'nicht-anonymisieren' einer Entität schwerer wiegt als das fälschlicherweise anonymisieren einer Entität, die nicht hätte anonymisiert werden müssen. Die niedrige FPR auf der anderen Seite lässt darauf schließen, dass die Systeme fast keine Entität der Klasse 'O' fälschlicherweise anonymisieren. Aus dem Bild fällt hierbei KSystem: Es anonymisiert mit einer FPR von 4% fälschlicherweise deutlich mehr Entitäten als die anderen Systeme. In absoluten Zahlen fällt diese Diskrepanz noch stärker auf: Es anonymisiert fälschlicherweise rund 8500 'O'-Tokens bei insgesamt 9046 zu anonymisierenden Entitäten im Testdatensatz. Des weiteren 'verpasst' es mit einem Recall (TPR) von 57% mehr als 40% aller zu anonymisierenden Entitäten. Während sich dies in Teilen darauf zurück führen lässt, dass das System gar nicht darauf ausgelegt ist, einzelne Klassen wie 'ROOM' oder 'CITATION' (Kumulierte Vorkommen im Testdatensatz: 358 Tokens, näheres in Sektion 4.4.2) zu anonymisieren, scheint es im Allgemeinen Probleme mit unregulären Daten dieser Art zu haben. Die Gründe sowie die Ausprägungen dieser Problematik werden in den späteren Abschnitten genauer analysiert.
Bezüglich der restlichen Systeme, welche sich in der oberen, linken Ecke gruppiert hatten, lohnt sich ein näherer Blick (Abbildung 19). Diese zeigen mit einer FPR zwischen 0.0005 und 0.0025 deutlich schwächere Tendenzen, Entitäten fälschlicherweise zu anonymisieren (zwischen 107 und 535 Tokens im Testdatensatz). Das LCRF weist hierbei hier die niedrigsten Werte auf. Trotz der geringen FPR zeigen die

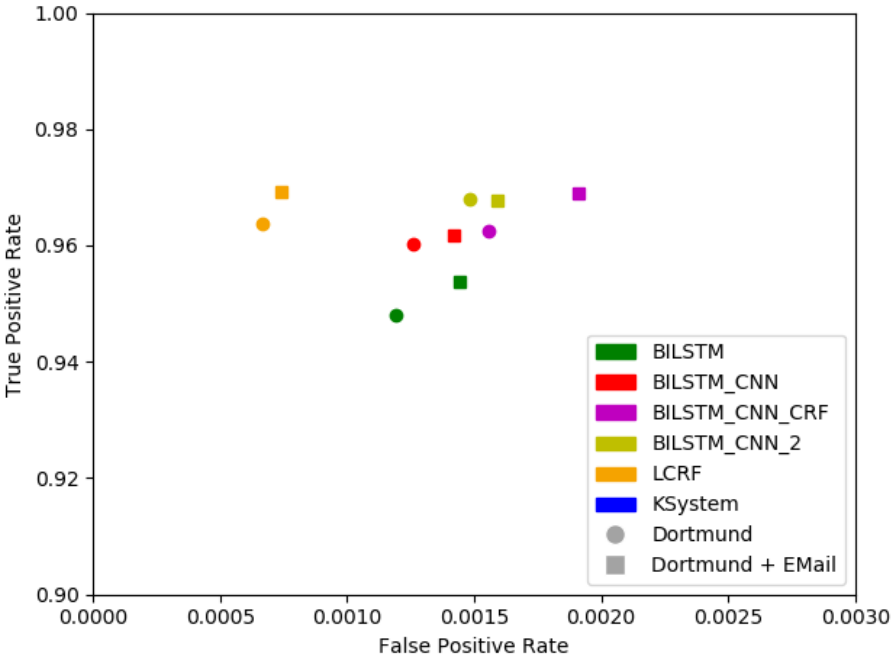


Abbildung 19: Ein Ausschnitt des ROC-Spaces mit den Leistungen der ML-Systeme auf dem Dortmund Chat Korpus

Systeme hohe Recall-Werte zwischen 94% sowie 97%. Besonders gut schneidet auch hierbei das LCRF ab. Auch das BILSTM_CNN_2 sowie BILSTM_CNN_CRF_COMP erreichen sehr hohe Werte, doch diese nehmen dafür eine höhere FPR in Kauf. Im Falle einer Anonymisierung ist aber, wie oben erläutert, davon auszugehen, dass das fälschlicherweise nicht-anonymisieren einer Entität schlechter ist, als eine nicht zu anonymisierende Entität zu anonymisieren.

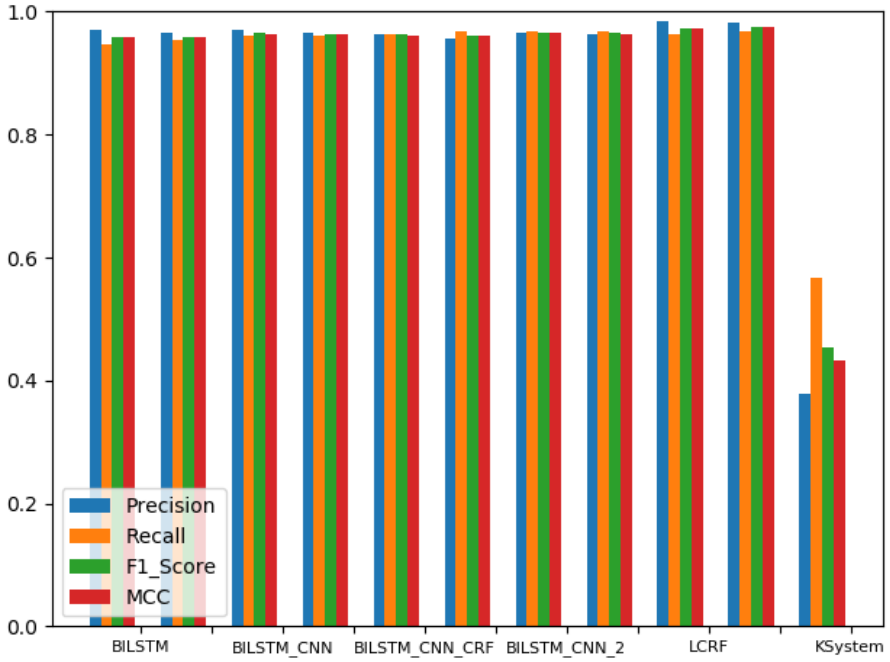


Abbildung 20: Metriken für die Binäre Klassifikation auf dem Dortmund Chat Korpus - die jeweils rechte Gruppe von Säulen bezieht sich auf das entsprechende '_COMP' System

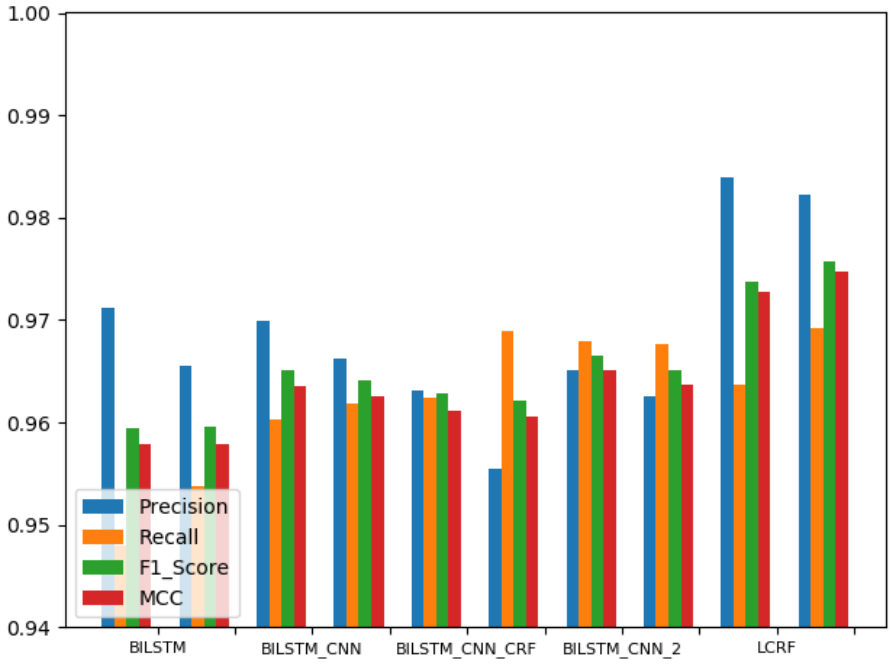


Abbildung 21: Ein Zoom auf die Metriken für die Binäre Klassifikation auf dem Dortmund Chat Korpus - die jeweils rechte Gruppe von Säulen bezieht sich auf das entsprechende '_COMP' System

Dementsprechend ist ein hoher Recall stärker zu gewichten als eine niedrige FPR.
Im Allgemeinen (abgesehen von BILSTM_CNN_2) erhöht das Training auf dem kompletten Datensatz sowohl den Recall, als auch die FPR. Besonders im Falle des LCRF geschieht dies in einem besonders günstigen Verhältnis. Doch auch bei den anderen Systemen, abgesehen von BILSTM_CNN_2, erhöht sich der Recall in einer stärkeren Rate als die FPR (durch die Skalierung der Grafik kann dies fälschlicherweise anders wirken). Daraus folgend ist das Training auf beiden Datensätzen trotz deren unterschiedlichen Aufbaus für die binäre Evaluation vorteilhaft, da wie oben erwähnt ein hoher Recall eine wichtigere Rolle spielt.
Ein Blick auf weitere, binäre Metriken bietet ein ähnliches Bild wie Abbildung 20 zeigt. Auch hier setzten sich die restlichen Systeme deutlich von KSystem ab, wobei besonders die niedrige Precision von KSystem ins Auge fällt. Für einen genaueren Vergleich der anderen Systeme ist wiederum eine detailliertere Betrachtung der Spitzen notwendig (Abbildung 21). Dort lässt sich erkennen, dass, gemessen an F1-Score sowie MCC, das LCRF die besten Ergebnisse liefert, gefolgt von BILSTM_CNN_2. Interessant ist, dass das BILSTM_CNN von Ma et al. im Durchschnitt besser abschneidet als ihr System inklusive dem CRF. Einzig BILSTM_CNN_CRF_COMP zeichnet sich durch einen besonders hohen Recall aus. Im Allgemeinen zeigt sich aber die Addition eines CNN - sowohl im ROC als auch in diesen Metriken - als vorteilhaft, wie man an dem Vergleich zum BILSTM sehen kann. Des weiteren deckt sich die Beobachtung im ROC, dass das Training auf beiden Datensätzen den Recall zwar steigen lässt, die Precision aber im Gegenzug in ähnlichem Maße sinkt.
Für weitere Einblicke abseits der binären Klassifikation werden nun Metriken für die Multiklassen-Klassifikation betrachtet (Abbildung 22). Dabei wurde jeweils die 'Macro' Variante gewählt, da sie durch die äquivalente Gewichtung aller Klassen in diesem Fall ein differenzierteres Bild bietet (die Werte der 'Micro'-Metriken erreichen durch die deutliche Überlast an 'O'- sowie 'NICK'-Labeln fast alle 1). Zu beachten ist hierbei, dass bei dieser Grafik kein Zoom vorliegt und die Unterschiede in den Metriken daher optisch geringer ausfallen. Des weiteren wurde MCC in diesem Falle außen vor gelassen, da alle Systeme in diesem Setup sehr hohe Werte in dieser Metrik erreichen (begründet durch die großen Klassen 'NICK' sowie 'O').

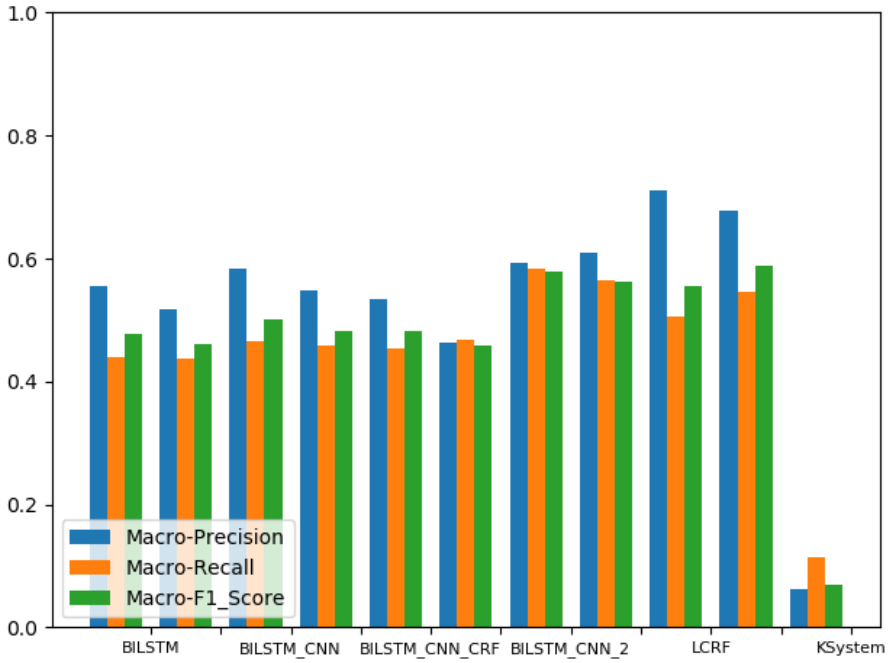


Abbildung 22: Metriken für die Klassifikation auf dem Dortmund Chat Korpus - die jeweils rechte Gruppe von Säulen bezieht sich auf das entsprechende '_COMP' System

Die Grafik macht ersichtlich, dass auch in der Multiklassen-Klassifikation das LCRF sowie BILSTM_CNN_2 die besten Ergebnisse erreichen. Auch die Beobachtung, dass das CRF als Addition zum BILSTM_CNN in diesem Szenario nicht vorteilhaft ist, bestätigt sich. Im Allgemeinen zeigen die Systeme durchweg deutlich schlechtere Ergebnisse. Dies ist damit begründet, dass die Systeme mit der Unterscheidung von, vor allem, kleineren Klassen Probleme haben (näheres dazu in Sektion 4.4.2), welche einen verhältnismäßig starken Einfluss auf die Micro-Metriken ausüben. Besonders scheint dies auf KSystem zuzutreffen, welches einen F1-Score nahe der 10% Marke erreicht. Doch dies lässt sich teilweise auf eine unterschiedliche Struktur in den Labels zurückführen. Dies wird in Sektion 4.4.2 näher behandelt. Während das Training auf beiden Datensätzen den Recall in der binären Klassifikation erhöht hat, ist dies hier nur noch für das LCRF sowie das BILSTM_CNN_CRF der Fall. Im Falle des BILSTM_CNN_2 sinkt der Recall etwas, aber die Precision erhöht sich. Die Gründe dafür werden in Sektion 4.4.2 genauer betrachtet.

E-Mail Korpus

Es konnten einige Erkenntnisse über die Leistungen der Systeme auf den unregulären Daten des Dortmund Chat Korpus gewonnen werden. Nun ist es Zeit, einen Blick auf den E-Mail Korpus zu werfen, welcher reguläre Daten beinhaltet. Auch hierbei wird mit der Betrachtung der binären Klassifikation begonnen.
Für diesen Korpus liegen nicht, wie es beim Dortmund Chat Korpus der Fall war, zwei Leistungsgruppen vor, sondern drei, wie Abbildung 23 zeigt. Denn die Systeme, welche ausschließlich auf dem Dortmund Korpus trainiert wurden, trennen sich von der Leistung hier klar von denjenigen, die auch auf dem E-Mail Korpus trainiert worden sind. Diese beiden Gruppen werden folgend getrennt in einer detaillierten Betrachtung analysiert. Vom Gesamtbild ausgehend fällt noch ins Auge, dass das KSystem auf dem E-Mail Korpus deutlich bessere Leitungen zeigt. Hier steigt die TPR von 57% auf 76% und die FPR fällt von 4% auf 3%. Dies ist damit zu erklären, dass das KSystem auf Texte solcher Art mit entsprechenden Informationen, wie Kreditkartennummern und ähnlichem, ausgelegt ist.

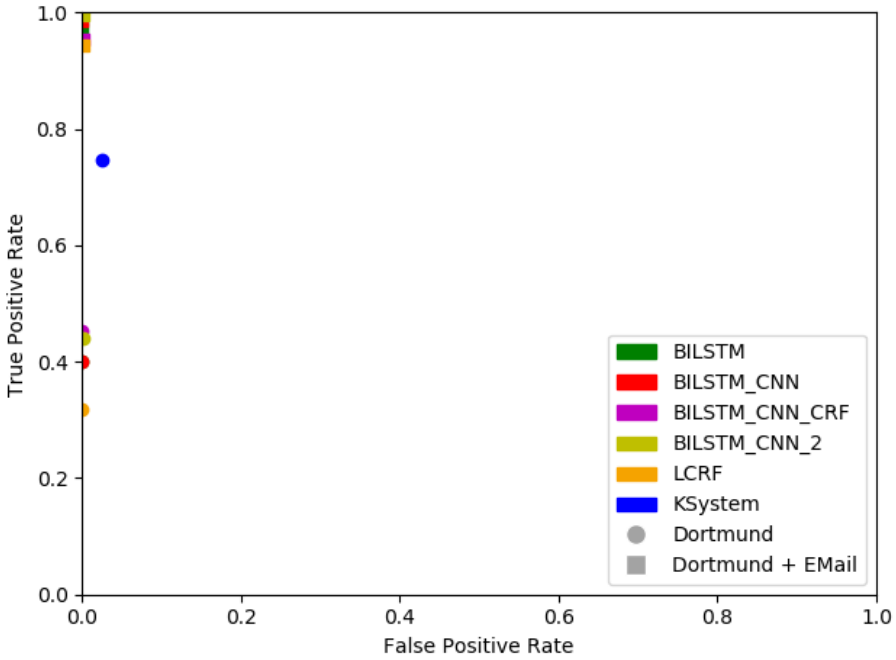


Abbildung 23: Die Leistungen der verschiedenen Systeme auf dem E-Mail Korpus im ROC-Space

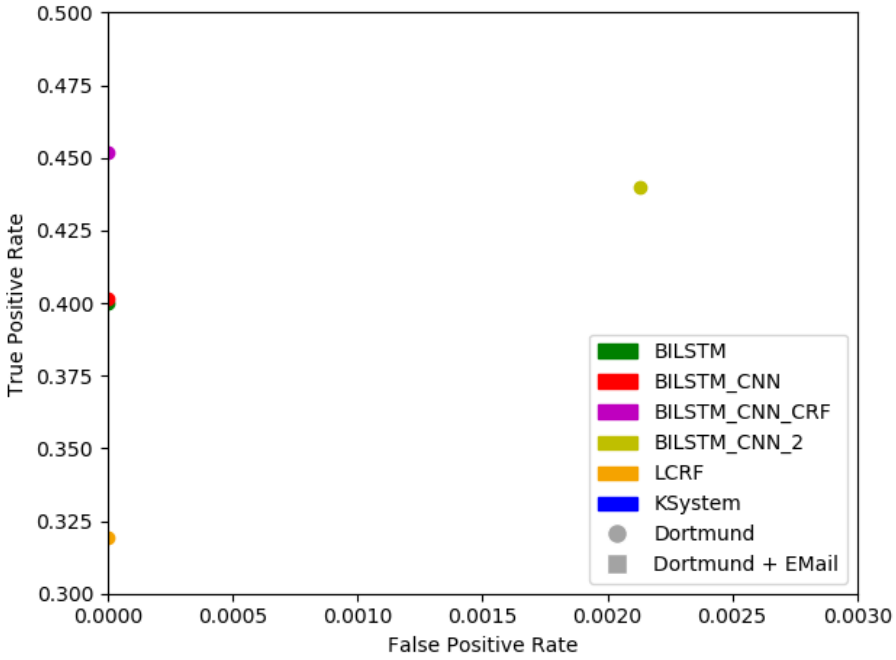


Abbildung 24: Ein Ausschnitt des ROC-Spaces mit den Leistungen der verschiedenen Systeme auf dem E-Mail Korpus im ROC-Space

Nichtsdestotrotz weist es gegenüber den '_COMP'-Systemen einen deutlichen geringeren Recall, sowie eine deutlich höhere FPR auf. Dies deckt sich mit den Leistungen des bauähnlichen Systems von Neamatullah et al., welches in Sektion 3.1 vorgestellt wurde.
Betrachtet man die untere Gruppe von Systemen (Abbildung 24), also diejenigen, die nicht auf dem E-Mail Korpus trainiert wurden, fällt die geringe FPR auf. Einzig BILSTM_2 klassifiziert Tokens der Klasse 'O' fälschlicherweise als eine der anderen Klassen, 4 (von 2176) sind es insgesamt. Im Allgemeinen

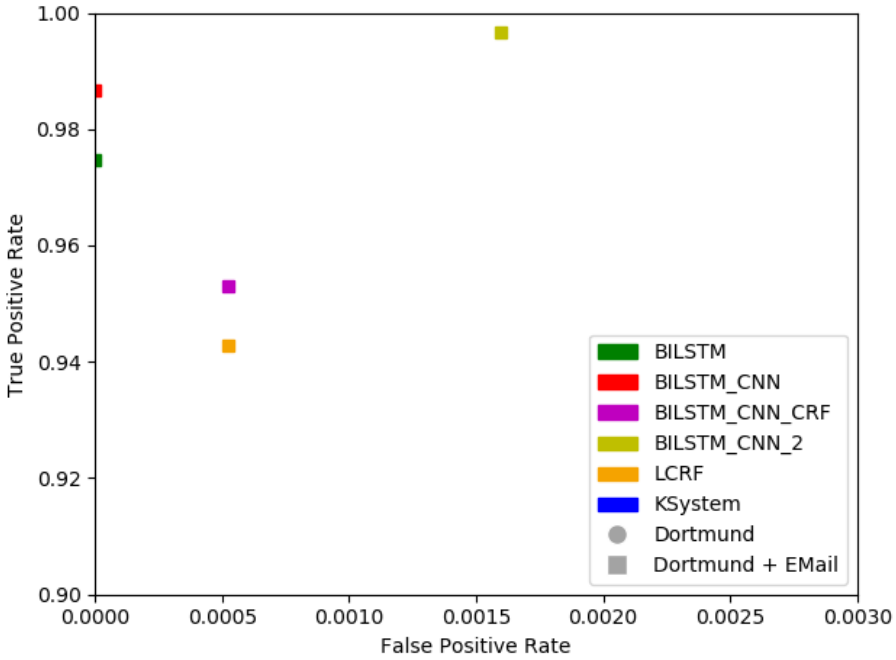


Abbildung 25: Ein Ausschnitt des ROC-Spaces mit den Leistungen der 'COMP'-Systeme auf dem E-Mail Korpus im ROC-Space

erreichen die Systeme aber nur einen Recall zwischen 30% und 45%. Dies lässt sich damit begründen, dass die Systeme sich hierbei in einem Out-of-Domain Test befinden: Sie konnten Texte dieser Art während des Trainings nicht kennen lernen. Auch bestimmte Ausprägungen von Entitäten, wie zum Beispiel Kreditkartennummern oder Bankdaten, sind den Systemen so unbekannt. Als Folge dessen sind insbesondere die Leistungen auf der Klasse 'NUMBER' schlecht (mehr dazu in Sektion 4.4.2). Nichtsdestotrotz sind sie in der Lage, die Transferleistung zwischen den Datensätzen zumindest eingeschränkt zu erbringen. Das LCRF, welches auf dem Dortmund Korpus die besten Ergebnisse erreichen konnte, schneidet in diesem Test mit einem Recall von 32% am schlechtesten ab. Hingegen erreicht das BILSTM_CNN_2, welches auch auf dem Dortmund Korpus gute Ergebnisse vorzeigen konnte, den zweitbesten Recall mit 44%, wodurch dessen etwas höhere FPR wenig ins Gewicht fällt. Das BILSTM_CNN_CRF, welches auf dem Dortmund Korpus vor allem im Mittelfeld aufzufinden war, schneidet in diesem Out-of-Domain Test mit einem Recall von 45% als Bester der Gruppe ab.

Im Falle der oberen Gruppe ('COMP'-Systeme, Abbildung 25) wiederholt sich das Bild der niedrigen FPR, welche auch im Vergleich zu den Ergebnissen auf dem Dortmund Korpus sehr gering ist. Zu beachten ist hierbei aber, dass es sich bei diesen kleinen Raten aufgrund der geringen Größe des Datensatzes nur um einzelne Tokens handelt. So klassifizieren zum Beispiel sowohl das BILSTM_CNN_CRF_COMP als auch das LCRF_COMP exakt ein falsch-positives, was zu ihrer FPR von 0.05% führt. Auch das BILSTM_CNN_2_COMP klassifiziert nur zwei positive Beispiele fälschlicherweise als negativ, was zu dessen hohen Recall von fast 100% führt. Dementsprechend ist bei diesen Werten die potentielle Varianz hoch. Im Allgemeinen sind die Ergebnisse aller Systeme sehr gut. Mit Ausnahme vom BILSTM_CNN_CRF_COMP sowie LCRF_COMP sind sie ebenso besser als auf dem Dortmund Chat Korpus. Dies ist auch von daher gehend interessant, da die Systeme trotz der Tatsache, dass der Dortmund Korpus fast 200 mal so viele Tokens wie der E-Mail Korpus zum Trainingssatz beitrug, sie die speziellen Anforderungen des E-Mail Korpus (wie zum Beispiel Kreditkartennummern) gut umsetzen. Eine Ausnahme bilden hierbei das LCRF sowie BILSTM_CNN_CRF_COMP, welches sogar schlechter abschneidet als die Version des Systems ohne die CNN- sowie CRF-Additionen (BILSTM_COMP). Sie Beide scheinen mit der wechselnden Struktur in den Daten schlechter umgehen zu können als die anderen Systeme.

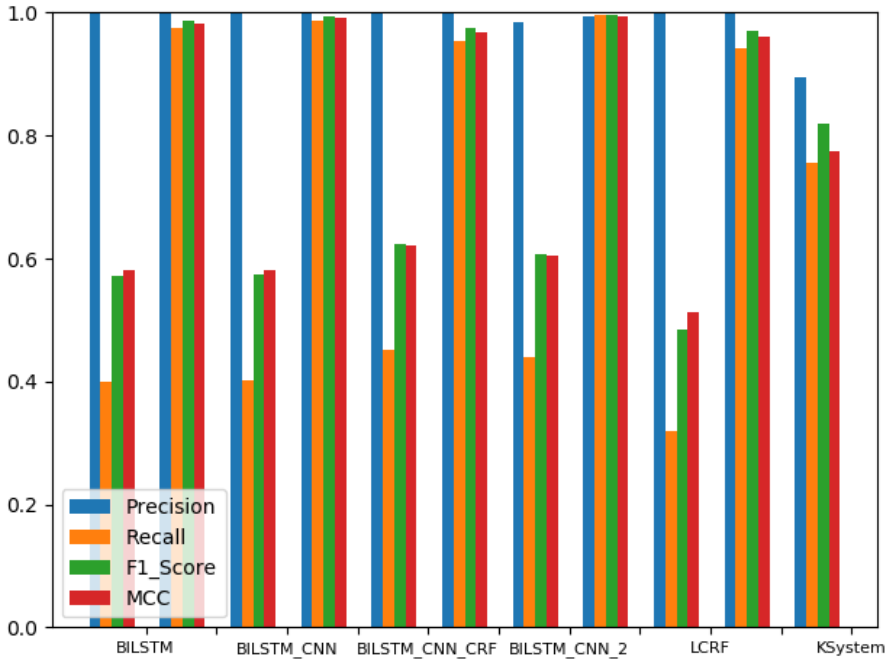


Abbildung 26: Metriken für die Binäre Klassifikation auf dem E-Mail Korpus - die jeweils rechte Gruppe von Säulen bezieht sich auf das entsprechende '_COMP' System

Ein Blick auf die binären Metriken (Abbildung 26) bestätigt die eben erlangten Erkenntnisse, zum Beispiel dass das BILSTM_CNN_2_COMP die besten Gesamtleistungen erbringt (gemessen an F1-Score sowie MCC). So spiegelt sich die niedrige FPR in einer sehr hohen Precision über alle Systeme wider, auch bei KSystem ist sie im Vergleich sehr hoch. Das bestätigt die Annahme, dass es im E-Mail Korpus durch die reguläre Struktur einfacher für die Systeme ist, 'zu anonymisieren' von 'nicht zu anonymisieren' zu unterscheiden. Dies trifft insbesondere auf die Systeme, die auf diesem Korpus trainiert wurden, zu.

Für die Analyse der Multiklassen-Klassifikation (Abbildung 27) wurden analog zum Dortmund Chat Korpus die 'Macro'-Metriken verwendet, um eine vergleichende Evaluation zu ermöglichen. Der MCC wurde in diesem Fall inkludiert, da die relative Häufigkeit von 'O' deutlich geringer ist und keine Klasse der Größe von 'NICK' in diesem Korpus existiert. Während sich auch hier einige Erkenntnisse aus der binären Klassifikation bestätigen, wie zum Beispiel das gute Abschneiden des BILSTM_CNN_2_COMP sowie des BILSTM_CNN_CRF-Systems, schneidet das LCRF_COMP im Vergleich besser ab als in der binären Klassifikation. Es erreicht sowohl einen höheren F1-Score als auch einen höheren MCC als BILSTM_CNN_CRF_COMP und einen höheren F1-Score als BILSTM_CNN_COMP, während dessen MCC hingegen höher bleibt. Letzteres rührt daher, dass das LCRF zwar quantitativ mehr Fehler macht als das BILSTM_CNN_COMP, aber in kleineren Klassen besser abschneidet. Da der MCC die Größe der Klassen in die Gewichtung mit einbezieht, der Macro-F1-Score hingegen alle Klassen gleich gewichtet, kommt diese Differenz zu Stande. Des weiteren fällt, analog zum Ergebnis auf dem Dortmund Chat Korpus, auf, dass das KSystem im Vergleich zu dem Ergebnis in der binären Klassifikation (abgesehen von MCC) sehr schlecht abschneidet. Dies ist besonders auf kleineren Klassen der Fall (daher der vergleichsweise hohe MCC-Wert). Diese Problematik wird in Sektion 4.4.2 näher betrachtet.

Zusammenfassung

Die Analyse der Gesamtergebnisse hat gezeigt, dass es sehr gut möglich ist, Systeme aus der NER in dem Bereich der Anonymisierung einzusetzen, auch auf den unregulären Daten des Dortmund Chat Korpus. Vor allem im Bereich der binären Klassifikation, welcher für die rechtliche Sicherheit der Anonymisierung besondere Relevanz besitzt, liefern die ML-Systeme sehr gute Leistungen und übertreffen in

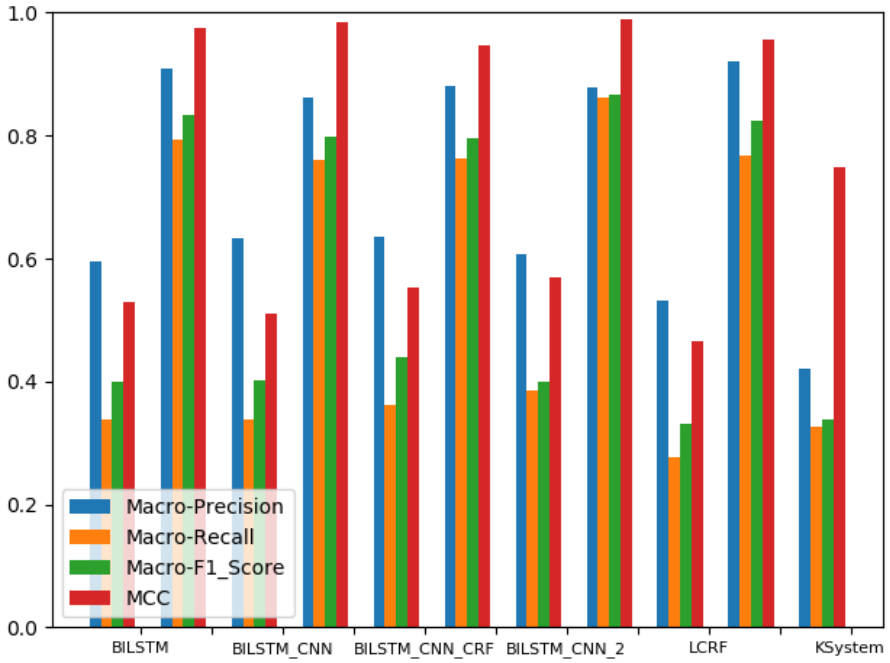


Abbildung 27: Metriken für die Klassifikation auf dem E-Mail Korpus - die jeweils rechte Gruppe von Säulen bezieht sich auf das entsprechende '_COMP' System

beiden Datensätzen die Leistungen des Vergleichssystems aus der Industrie. Einzig in einem Szenario, in welchem E-Mails zu anonymisieren sind ohne dass entsprechende Trainingsdaten vorliegen, wäre die Anwendung KSystems von Vorteil.
Im Rahmen der Multiklassen-Klassifikation hingegen zeigen sich deutlich schlechtere Ergebnisse. Diese sind zwar für den unmittelbaren Erfolg der Anonymisierung nicht relevant, doch der Wert des anonymisierten Textes für weitere Verarbeitungen steigt mit einer richtigen Klassifikation der anonymisierten Entitäten deutlich (siehe Sektion 4.4.2). Daher werden in der folgenden Sektion die Ursachen für die schlechtere Leistung im Multiklassen-Szenario untersucht. Auch wird ein Blick darauf geworfen, welche Klassen für die Systeme besonders leicht und welche besonders schwer zu detektieren waren.
Im Allgemeinen scheint das Training auf beiden Datensätzen für beide Testdatensätze vorteilhaft zu sein - nicht nur für den Test auf dem E-Mail Korpus, wodurch sich die Systeme nicht mehr in einem Out-of-Domain Szenario befinden. Auch für den Test auf dem Dortmund Chat Korpus führt die erhöhte Menge an Trainingsdaten, sowie die Addition neuer Arten von Entitäten, zu einem verbesserten Recall in der binären Klassifikation. Im Gegenzug sinken zwar Precision sowie die Leistungen in der Multiklassen-Klassifikation, doch dies ist im Szenario der Anonymisierung aus genannten Gründen weniger stark zu gewichten.
Bezüglich der Leistung der verschiedenen Systeme hat sich besonders das BILSTM_CNN_2 hervorgetan, welches auf beiden Datensätzen in beiden Ausführungen sehr gute Ergebnisse erreichte. Dies rührt wohl auch daher, dass dieses System im Vergleich zu den anderen Systemen einen höheren Recall stärker gewichtet als eine niedrige FPR / hohe Precision, was im Szenario der Anonymisierung vorteilhaft ist. Interessant ist dies auch vor dem Hintergrund, dass das System auf dem GermEval Datensatz von einem vergleichsweise geringen F1-Score von 68% berichtete (siehe 3.2.2).
Die Systeme von Ma et al. lagen mit ihren Leistungen meist im Mittelfeld. Während sich die Addition eines CNNs (BILSTM_CNN) zu dem BILSTM durchweg als vorteilhaft gezeigt hat, hat die weitere Addition der CRF-Komponente (BILSTM_CNN_CRF) zwar zu besseren Out-of-Domain Ergebnissen geführt, aber an anderen Stellen die Leistungen verschlechtert. Dies ist interessant, da die Addition des CRF einen deutlichen Leistungsgewinn in der NER brachte.

Das LCRF zeigte auf dem Dortmund Chat Korpus besonders gute Ergebnisse, während es auf dem E-Mail Korpus am schlechtesten von allen ML-Systemen abgeschnitten hat, insbesondere im Out-of-Domain Test. Die Gründe für diese Diskrepanz werden in der nächsten Sektion genauer analysiert.

4.4.2 Klassen-Orientierte Evaluation

Eine reine Anonymisierung des Textes, wie es die binäre Klassifikation vorsieht, ist aus rechtlicher Sicht vollkommen ausreichend. Doch wendet man eine Anonymisierung an, hat man sich bewusst für eine Anonymisierung und gegen eine Löschung der Daten entschieden, welche rechtlich ebenso konform wäre. Die Motivation hinter einer Anonymisierung ist in der Regel, aus dem anonymisierten Text noch nutzen ziehen zu können - auch, wenn alle personenbezogenen Daten daraus entfernt wurden. Hierfür bietet eine Kategorisierung der anonymisierten Entitäten, wie es durch die Multiklassen-Klassifikation der Fall ist, deutliche Vorteile: Denn auch nach der durchgeführten Anonymisierung ist erkennbar, ob an einer gewissen Stelle zum Beispiel ein Städtename (GPE) oder doch ein konkreter Ort (LOC) stand [53]. Mit solch einer Kategorisierung ist es auch möglich, den Text durch eine Wiedereinsetzung, wie es in dieser Arbeit für den Dortmund Chat Korpus durchgeführt wurde, wieder in eine vollständige Form zu überführen. Daher beschäftigt sich dieser Abschnitt mit einem genaueren Blick auf die Leistungen der Systeme in den verschiedenen Klassen: Warum erreichen die Systeme in der binären Klassifikation deutlich höhere Ergebnisse als in der Multiklassen-Klassifikation? Auf welchen Klassen schlagen sich die verschiedenen Systeme besonders gut, auf welchen eher schlecht?

Durchschnittliche Leistungen in den verschiedenen Klassen

Um eine Basis für letztere Fragestellung zu legen, gilt es erst einmal zu evaluieren, welche Klassen über alle Systeme hinweg besonders gut, welche besonders schlecht erkannt wurden. Darüber bietet Abbildung 28 eine Übersicht. Sie zeigt den durchschnittlichen F1-Score pro Klasse über alle Systeme hinweg. Die jeweils linke Gruppe von Säulen zeigt die Leistung auf den 'B-'-Tags, die rechte Gruppe die Leistung auf den jeweiligen 'I-'-Tags. Zu beachten ist, dass der E-Mail Korpus keine Entitäten der Typen 'NICK',

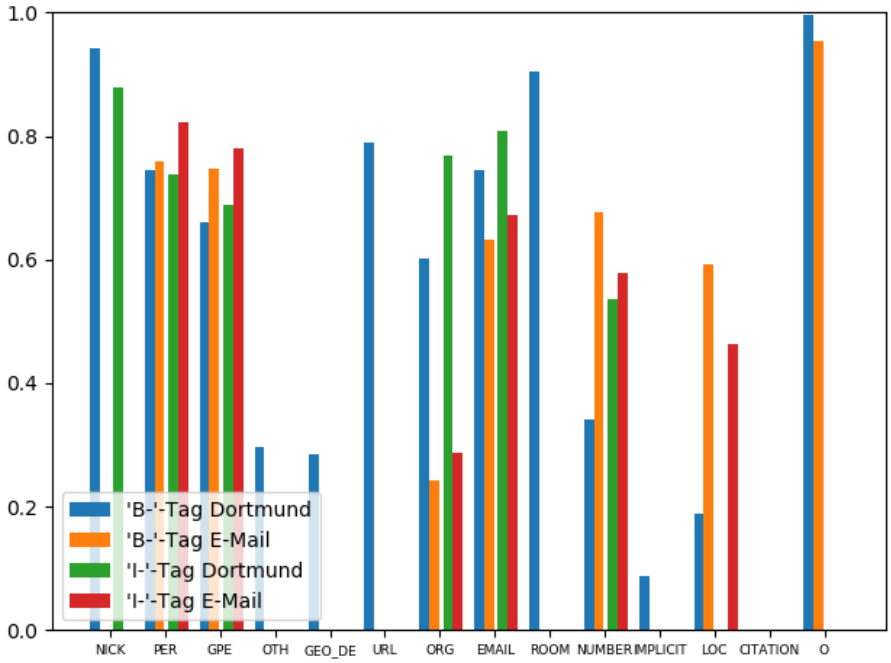


Abbildung 28: Durchschnittlicher F1-Score aller Systeme in den verschiedenen Klassen - Die jeweils linke Gruppe von Säulen zeigt die Leistung auf den 'B-'-Tags, die rechte Gruppe die Leistung auf den jeweiligen 'I-'-Tags

'OTH', 'GEO_DE', 'ULR', 'ROOM', 'IMPLICIT' sowie 'CITATION' enthält. Im Dortmund Korpus hingegen kommen keine Entitäten des Typs 'I-GEO_DE', 'I-URL', 'I-ROOM' sowie 'I-LOC' vor. Dementsprechend liegen für diese Klassen keine Werte vor.
Die Grafik gewährt den Einblick, dass sich die Leistungen auf den verschiedenen Klassen über die beiden Datensätze hinweg meist sehr ähneln, wobei zu beachten ist, dass der Durchschnitt des E-Mail Korpus durch die Out-of-Domain Tests vergleichsweise geringer ist. Ausnahmen hiervon bilden 'ORG', 'NUMBER', sowie 'LOC'. Bei 'ORG' sind die Daten aufgrund sehr geringer Frequenzen im E-Mail Korpus nicht zuverlässig: Der Testdatensatz besitzt gerade einmal 2 'B-'- sowie 2 'I-'-Tags. Daher können für diese Klasse keine zuverlässigen Analysen getroffen werden. Bei 'NUMBER' fällt besonders die deutlich bessere Leistung der Systeme auf den 'B-'-Tags des E-Mail Datensatzes auf, während sich die Leistung in den 'I-'-Tags sehr ähnelt. Dies kann man darauf zurück führen, dass Nummern im E-Mail Korpus sehr eindeutig eingeleitet werden, zum Beispiel durch "Vertrag mit der Nummer..." oder 'Kreditkartennummer: ...". Dies scheint es den Systemen deutlich zu vereinfachen, den Beginn einer Nummer auszumachen. Ähnliches gilt auch für 'B-LOC': Dies taucht im Rahmen des E-Mail Korpus ausschließlich in dem sehr regulären Rahmen von Adressen auf, während es sich im Dortmund Chat Korpus an diversen, schwerer zu detektierenden Stellen wie diesen hier befinden:

wäre jetzt beinahe in die Bergstraße (B-LOC) gefahren
. . .
und kennsch du Ischgl (B-LOC) ?

An ersterer Stelle könnte ebenso ein 'ORG'-Tag platziert sein, wie zum Beispiel 'TU' (für TU-Darmstadt), an zweiterer zum Beispiel ein Name (PER oder NICK).
Im Allgemeinen zeigt sich die Detektion von 'B-'-Tags schwieriger, insbesondere bei der Klasse 'NUMBER' im Dortmund Korpus. Ein möglichen Grund dafür bietet die Tatsache, dass 'I-'-Tags häufig in Gruppen mit anderen 'I-'-Tags oder mindestens einem 'B-'-Tag auftauchen. 'B-'-Tags hingegen kommen häufig auch alleine vor, wodurch es für die Systeme so schwerer sein kann, diese richtig zu detektieren. Dies zeigt sich zum Beispiel im Dortmund Chat Korpus in den Klassen 'NICK' sowie 'GPE', für welche der Korpus deutlich mehr 'B-'- als 'I-'-Tags enthält.
Bei einer Betrachtung über die Klassen hinweg zeigen sich besonders gute Leistungen in den Klassen 'O', 'NICK' sowie 'ROOM'. Gerade bei 'O' ist dies auf die sehr hohen Frequenzen dieser Klasse zurück zu führen. Im E-Mail Korpus ist der F1-Score bedingt durch die Out-of-Domain Tests niedriger. Ähnliches gilt für die Klasse 'NICK', welche die höchste Frequenz aller anderen Klassen aufzuweisen hat. Des weiteren kommen Nicknamen häufig an leicht zu detektierenden Stellen vor, wie zum Beispiel in Adressierungen ('@NICK' oder 'an NICK:') oder bei Systemnachrichten ('NICK hat den Raum Betreten'; 'NICK hat den Raum verlassen'). Auch die Satzzeichen am Ende von Nicknamen, wie sie in ca. 25% aller Fälle auftreten, stellt die Systeme vor keine größeren Schwierigkeiten. Einzig der etwas geringere F1-Score auf I-Tags lässt darauf schließen, dass es zumindest manchmal etwas problematisch ist. Die guten Ergebnisse auf 'PER' sowie 'NICK' decken sich weiterhin mit Erkenntnissen aus verwandten Arbeiten der NER (siehe Sektion 3.2.2), über welche sich abzeichnete, dass besonders in den Kategorie 'PER' gute Leistungen erreicht werden. Gute Leistungen auf der Klasse 'LOC' hingegen spiegeln sich hier nur im Falle 'GPE' (welche auch als Teil der NER-Kategorie 'LOC' zu sehen ist) wider. Ergebnisse auf der Klasse 'LOC' sind, besonders auf dem Dortmund Chat Korpus, vergleichsweise schlecht. Dies kann darauf zurück zu führen sein, dass 'LOC' im Rahmen der NER in deutlich regulären Kontexten aufzufinden ist, als es beim Dortmund Chat Korpus der Fall ist.
Während die Klasse 'ROOM' zwar deutlich weniger Vorkommen als 'NICK' aufzuweisen hat, ist es die nächst-häufigste Klasse. Auch sie taucht besonders häufig in Systemnachrichten auf und ist durch diesen regulären Kontext leichter zu detektieren. Zu beachten ist, dass dadurch, dass KSystem nicht für die Detektion solcher Raumnamen ausgelegt ist (und dementsprechend keine davon getaggt hat) und es dadurch den F1-Score auf 'ROOM' um ca. 8% verringert (für die Leistungen der jeweiligen Systeme, siehe Abbildung 29). Besonders schlecht hingegen sind die Leistungen auf den Klassen 'OTH', 'GEO_DE',

’IMPLICIT’ sowie ’CITATION’. ’OTH’ sowie ’IMPLICIT’ zeichnen sich dabei sowohl durch sehr diverse Entitäten, als auch einen stark wechselnden Kontext aus, wie auch folgende Beispiele aus dem Korpus demonstrieren:

gets denn in zuekonf au i dim club "Nova (B-OTH)"detroit verastaltige?
. . .
Ist dein Freund auch Hamburgfan (B-OTH)
. . .
nen buch von Tolkin (B-IMPLICIT)
. . .
ich studiere Mathe (B-IMPLICIT) und Physik (B-IMPLICIT)

Diese unregelmäßigen Strukturen bereiten den Systemen Probleme. Insbesondere so, dass fast kein ’I-’-Tag der jeweiligen Klassen korrekt detektiert wurde. Ähnlich stellt sich dies bei ’GEO_DE’ dar. Während es zwar im Falle dieser Klasse keine ’I-’-Labels gibt und die konkrete Ausprägung der Entitäten regelmäßiger ist (Frankfurter, Darmstädter, . . .), bietet deren Kontext meist auch keine wieder-findbaren Strukturen:

hab grade gesehen das Konrad (B-NICK) Österreicher (B-GEO_DE) wie ich bin
. . .
Wilfried (B-NICK) fühlt O sich mit so vielen Schwaben (B-GEO_DE) nicht mehr wohl

Einen besonderen Fall stellt ’CITATION’ dar: Mit nur einem dreimaligem Vorkommen (’B-’-Tags) im Trainingsdatensatz sowie nur einem Vorkommen im Testdatensatz (das folgende Beispiel), ist es die am Schwächsten vertretene Klasse. Dies kombiniert mit der Tatsache, dass auch in diesen Fällen der Kontext sehr wechselhaft ist, hat dazu geführt, dass keines der Systeme einen Tag richtig erkannt hat.

muss mir noch nen knalligen titel einfallen lassen, aber etwa so:
[\:\-]
”Analyse (B-CITATION) von (I-CITATION) Wirtschaftskreisläufen (I-CITATION) in (I-CITATION) despotischen (I-CITATION) Regimen (I-CITATION)”
oder so ähnlich

Die Leistungen auf ’LOC’ sowie ’NUMBER’ wurden bereits im oberen Abschnitt behandelt.

Leistungen der Systeme auf den verschiedenen Klassen

Die vorherige Sektion hat einen Überblick darüber gegeben, welche Klassen grundlegend einfacher und welche schwerer für die Systeme zu detektieren waren. Dies bildet nun die Basis um die Fragestellung zu beantworten, welche Systeme sich auf welchen Klassen besonders gut, beziehungsweise schlecht schlagen. Genauere Informationen hierfür bietet Abbildung 29, welche die durchschnittlichen F1-Scores (über beide Datensätze sowie ’B-’-und ’I-’-Tags hinweg) eines jeden Systems für jede Klasse bereit stellt. Zu beachten ist hierbei, dass die F1-Scores für Labels, welche nicht in den Datensätzen vorkamen (siehe oben), bei der Berechnung bewusst nicht berücksichtigt wurde, um die Ergebnisse nicht zu verfälschen.
Im Allgemeinen bestätigt die Abbildung die Erkenntnisse der letzten Sektion. Zu erkennen ist zum Beispiel, dass fast alle Systeme auf der Klasse ’ROOM’ einen F1-Score von fast 1 erreichen und der in der letzten Sektion präsentierte Durchschnitt durch KSystem negativ beeinflusst wurde. Ähnliches gilt für die Klasse ’URL’, wo fast alle Systeme, mit Ausnahme von KSystem, BILSTM sowie BILSTM_COMP, sehr gute Ergebnisse erreichten. Gerade in dieser Kategorie scheinen sich Zeichen-basierte Features durch die Addition eines CNNs auszuzahlen. Auf die Leistungen von KSystem wird am Ende dieses Abschnitts näher eingegangen, zu Beginn werden die ML-Systeme analysiert.
Sowohl bei ’PER’, ’GPE’, ’ORG’ als auch bei ’NUMBER’ sowie ’LOC’ fächern sich die Leistungen der ML-Systeme in 2 Gruppen auf, wobei sich die obere Gruppe aus den ’COMP’-Systemen, die untere Gruppe aus den anderen Systemen bildet. Dies ist auch darauf zurück zu führen, dass die ’COMP’-Systeme auf dem E-Mail Korpus deutlich bessere Ergebnisse als ihre Pendants erzielt haben. Alle anderen Kategorien,

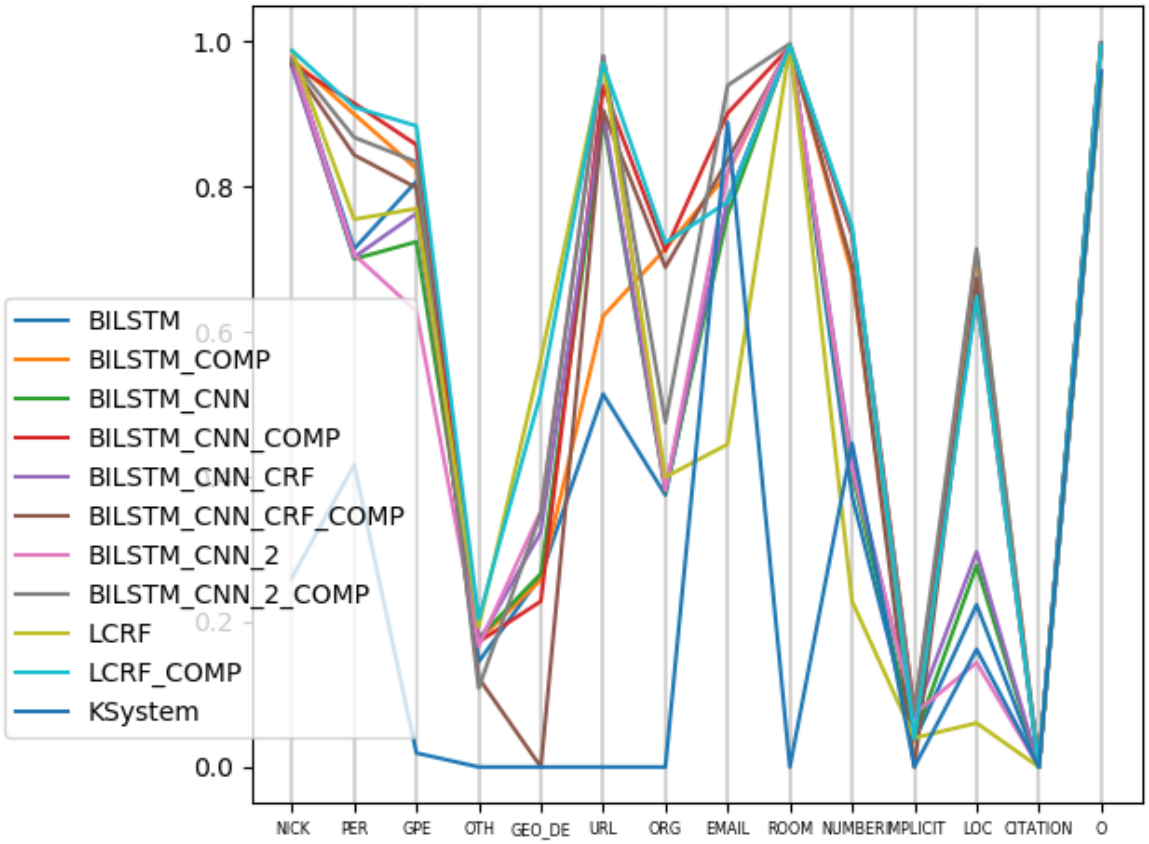


Abbildung 29: F1-Scores der Systeme auf den verschiedenen Klassen

in welchen die 'nicht-COMP'-Systeme ähnliche Leistungen wie ihre Pendants erreichen, kommen im E-Mail Korpus nicht vor und sind dementsprechend in den Werten nicht enthalten. Eine Ausnahme davon bildet (neben 'O') die Klasse 'EMAIL', in welcher die beiden Gruppen sehr ähnlich abgeschnitten haben. Das Konzept der Klasse 'EMAIL' ist für die Systeme also einfacher über Datensätze hinweg zu übertragen als in anderen Kategorien, wie zum Beispiel 'PER'. Um noch einen Endruck zu gewinnen, wie sich die Systeme ohne diesen Einfluss schlagen, wird später noch auf die Ergebnisse eingegangen (Abbildung 30), die ausschließlich auf dem Dortmund Chat Korpus erfasst wurden.
Bei der Gruppe der 'nicht-COMP'-Systeme gibt es keinen klaren Sieger: Während einige Systeme in manchen Kategorien die anderen überflügeln, bieten sie in den anderen Kategorien schlechte Ergebnisse. Ein Beispiel hierfür ist das BILSTM_CNN_CRF, welches in 'LOC' die beste Leistung einbringt, bei 'GPE' im Mittelfeld aufzufinden ist aber bei 'PER' sowie 'ORG' mit die schlechtesten Leistungen der Systeme erbringt.
In 'GEO_DE' fallen zwei Sachen ins Auge: BILSTM_CNN_CRF_COMP, welches sonst gute Leistungen erbringt, sagt keine Instanz der Klasse 'GEO_DE' vorraus und fällt als Folge dessen auf einen F1-Score von 0%. Sein Pendant, welches nur auf dem Dortmund Chat Korpus trainiert wurde, befindet sich hingegen im Mittelfeld. Die zusätzlichen Trainingsdaten, welche keine Instanz des Types 'GEO_DE' enthielten, haben also dazu geführt, dass das System diese Klasse gar nicht mehr vorraus sagt. Auf der anderen Seite schneiden beide LCRF-Systeme in dieser Kategorie im Vergleich außerordentlich gut ab. Im Falle dieser Architektur scheint es von Vorteil zu sein, dass viele Entitäten dieser Klasse ähnliche Endungen aufweisen. Bei den anderen Systemen, bei denen in dieser Kategorie durch die Addition des CNNs auf Zeichenebene eine bessere Leistung gegenüber dem BILSTM erwarten würde, trifft dies nur eingeschränkt zu.
Interessant an der Klasse 'EMAIL' ist, dass das LCRF trotz einiger E-Mail Vorkommen im Dortmund Korpus

mit Abstand am schlechtesten abschneidet. Die 'COMP'-Version hingegen bewegt sich im Mittelfeld aller Systeme. Dies bestätigt die Ergebnisse von oben, wonach das LCRF nicht so gut wie die anderen Systeme in der Lage ist, das Konzept dieser Klasse zwischen den Datensätzen zu übertragen, ähnliches gilt für die Kategorie 'NUMBER'. Die Klasse 'EMAIL' hingegen wird außerordentlich gut von BILSTM_CNN sowie BILSTM_CNN_2 erkannt. Die Addition eines CRFs verschlechtert die Ergebnisse hingegen.
Im Allgemeinen liefern LCRF_COMP sowie BILSTM_CNN_2_COMP, welche auch in den vorhergehenden Analysen die besten Ergebnisse erreichten, sich in den meisten Kategorien ein enges Rennen um die besten Werte. Eine Ausnahme hiervon bildet die Kategorie 'ORG', in welcher beide Versionen des BILSTM_CNN_2 deutlich schlechtere Leistungen als im Durchschnitt erbringen.
Es bleibt die Frage zu klären, warum dass das LSTM_COMP auf dem E-Mail Korpus vergleichsweise schlecht abgeschnitten hat, obwohl es, wie eben erläutert, auf fast allen Klassen sehr gute Leistungen zeigt. Dies lässt sich auf, im Vergleich zu den anderen Systemen, schlechte Leistungen in den Kategorien 'PER' sowie 'NUMBER' zurück führen. Während das LCRF_COMP in diesen Kategorien besonders auf dem Dortmund Korpus gut abschneidet (vergleiche Abbildung 30), liegt es in den Leistungen auf diesen Klassen im E-Mail Korpus zurück (vergleiche Tabelle 110 im Anhang). Es ist also schlechter als die anderen Systeme in der Lage, die unterschiedlichen Verwendungen dieser Klassen über die beiden Datensätzen hinweg zu erfassen.
Ein detaillierter Blick auf die Leistungen von KSystem offenbart, dass es sich besonders in den Kategorien 'EMAIL' sowie 'NUMBER' gut schlägt. Dies kann dadurch begründet werden, dass beide Kategorien über eine sehr regelmäßige Struktur verfügen (Bei 'NUMBER' zum Beispiel Konto- oder Telefonnummern). Unerwartet ist hingegen dass Ergebnis, dass das System trotz dieser sehr regulären Muster auch in diesen Kategorien nur im Mittelfeld der anderen Systeme aufzufinden ist. Es hat zum Beispiel Pro-

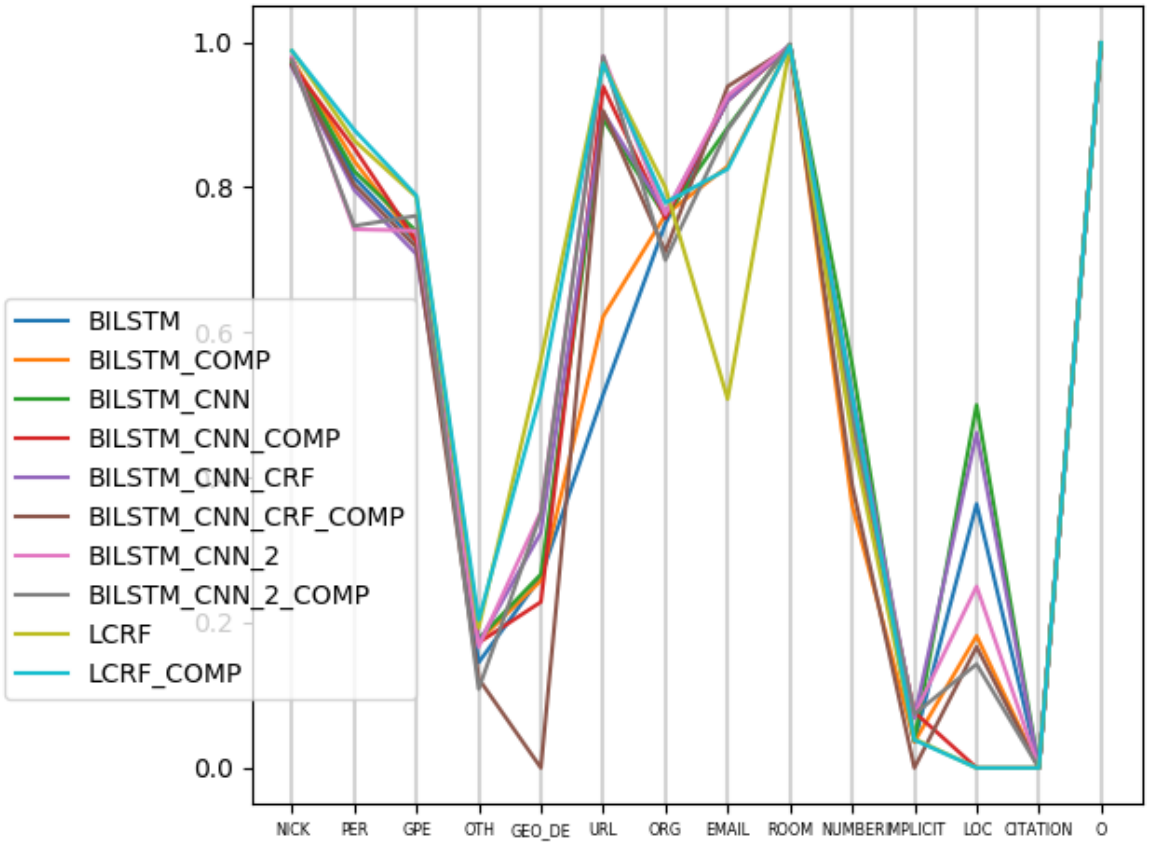


Abbildung 30: F1-Scores der Systeme auf dem Dortmund Chat Korpus, aufgetrennt in die verschiedenen Klassen

bleme, ungewöhnliche Emails zu detektieren. Kontonummern und Bankleitzahlen wie in den folgenden Beispielen, welche direkt dem Korpus entnommen wurden, anonymisiert es gar nicht.

08e8326@webchat.xs4all.ne
. . .
Kontonummer: 8903912971 Bankleitzahl: 61113387

Auch bei der Detektion von Orten befindet es sich eher im unteren Mittelfeld. Diese Probleme können, hingegen zu denen auf den restlichen Klassen, nicht auf Probleme mit unterschiedlichen Labels zurück geführt werden.
Abbildung 30 liefert eine analoge Ansicht, welche sich aber alleine auf die Ergebnisse des Dortmund Chat Korpus bezieht. Die Ergebnisse von KSystem wurden bewusst nicht inkludiert, da dessen Labels, wie oben beschrieben, besonders auf dem Dortmund Korpus eine andere Kategorisierung aufweisen. Hierbei ist zu erkennen, dass die oben beschriebene Trennung in 2 Gruppen (’COMP’ sowie ’nicht-COMP’) in diesem Falle nicht vorliegt. Auch zeigt sich, dass das Training auf beiden Datensätzen die Leistung auf bestimmten Klassen auch auf dem Dortmund Korpus erhöhen kann (zum Beispiel LCRF_COMP über alle Klassen hinweg), dies aber auch die Leistungen verschlechtern kann (zum Beispiel im Falle von BILSTM_CNN_COMP). LCRF_COMP, welches in der kombinierten Ansicht noch ein sehr gutes Ergebnis auf ’LOC’ erreichte, fällt hier auf einen F1-Score von 0%. Dies lässt darauf schließen, dass es zwar sehr gut in der Detektion von Orten auf dem E-Mail Korpus ist, dies aber nicht im Dortmund Korpus bieten kann. Im Allgemeinen bestätigen sich die meisten Erkenntnisse der letzten Grafik, zum Beispiel dass die Addition eines CNNs die Leistung auf der Kategorie ’URL’ deutlich verbessert. Die schlechten Leistungen einiger Systeme auf der Klasse ’ORG’ (wie zum Beispiel BILSTM_CNN_2_COMP) liegen hier nicht vor und lassen sich so auf den E-Mail Korpus zurück führen.

Verwechslungen zwischen Klassen

Nun gilt es, die andere der oben erwähnten Fragen zu beantworten: ”Warum erreichen die Systeme in der binären Klassifikation so viel höhere Ergebnisse?”. Dies läuft darauf hinaus, erst einmal herauszufinden, welche Klassen von den Systemen häufig verwechselt werden und so zu den schlechteren Ergebnisse führen. Dafür wird die in Tabelle 21 dargestellte Konfusionsmatrix zu Rate gezogen: Sie zeigt die durchschnittliche Fehlklassifikation aller ’COMP’-Systeme auf dem Dortmund Chat Korpus an. Es wurde bewusst darauf verzichtet, die anderen Systeme zu inkludieren, da die ’COMP’ Systeme die interessanteren der Modelle darstellen, da sie in der Lage sind, beide Datensätze gut zu anonymisieren. Des weiteren wurden auch die Ergebnisse des E-Mail Korpus nicht inkludiert, da alle ’COMP’-Systeme dort kaum Fehler in der Klassifikation gemacht haben. Am Ende der Sektion wird noch einmal kurz auf diese Systeme sowie auf den E-Mail Datensatz eingegangen.
Aus der Matrix wird direkt ersichtlich, dass die meisten Verwechselungen zwischen den Klassen mit der Klasse ’O’ geschieht (blau markiert). Häufiger ist dabei, vor allem bei den kleineren Klassen, dass zu anonymisierende Entitäten als ’O’ markiert werden (letzte Teile), als umgekehrt. Dies fällt zum Beispiel besonders in der Klasse ’IMPLICIT’ auf. Im Allgemeinen sind die relativen Häufigkeiten von Verwechslungen deutlich erhöht. Dies ist insbesondere der Fall bei Klassen, die im letzten Abschnitt als ’schwer zu detektieren’ klassifiziert wurden (zum Beispiel ’OTH’). Im Falle von ’NICK’ hingegen sind zwar die absoluten Häufigkeiten der Verwechslungen hoch, relativ gesehen zur Häufigkeit der Klasse sind dies jedoch geringe Werte. Dies bestätigt die Ergebnisse des letzten Abschnittes. Verwechslungen zwischen den ’zu anonymisierenden’-Klassen kommen deutlich seltener vor (magenta markiert). Vergleichsweise viele der Verwechslungen entstehen dadurch, dass verschiedene Klassen fälschlicherweise als ’NICK’ klassifiziert werden (erste Zeile). Das rührt wohl daher, dass die Systeme durch die sehr häufigen Vorkommen von ’NICK’ eher dazu tendieren, einen Token als ’NICK’ zu klassifizieren.

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6917.6**	0.0	4.8	2.8	3.4	0.0	1.4	0.0	0.0	0.0	1.0	0.0	0.2	0.0	0.4	0.2	0.2	0.0	0.2	0.0	0.0	0.0	0.2	0.0	0.0	0.0	109.4
I-NICK	0.0	**832.4**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	50.2
B-PER	9.4	0.0	**111.2**	0.6	0.2	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	24.8
I-PER	2.8	0.0	0.4	**109.6**	0.0	0.0	0.0	0.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	26.6
B-GPE	0.4	0.0	0.0	0.0	**136.0**	0.0	0.6	0.0	1.8	0.0	0.0	0.0	0.6	0.0	0.2	0.0	0.2	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	30.4
I-GPE	0.0	0.0	0.0	0.0	1.4	**45.4**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	14.2
B-OTH	0.0	0.0	0.0	0.0	0.0	0.0	**8.8**	0.0	0.0	0.0	0.0	0.0	5.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.6
I-OTH	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-GEO_DE	0.0	0.0	0.4	0.0	0.4	0.0	0.0	0.0	**4.6**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	3.4
I-GEO_DE	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-URL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**42.2**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.8
I-URL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-ORG	0.0	0.0	0.0	0.0	0.0	0.0	3.2	0.0	0.2	0.0	0.0	0.0	**40.6**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.8	0.0	0.0	0.0	11.0
I-ORG	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	**6.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-EMAIL	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**4.6**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4
I-EMAIL	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**41.2**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6
B-ROOM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**347.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0
I-ROOM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-NUMBER	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	**8.0**	1.0	0.0	0.0	0.4	0.0	0.0	0.0	10.4
I-NUMBER	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6	**12.4**	0.0	0.0	0.0	0.0	0.0	0.0	14.2
B-IMPLICIT	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**1.2**	0.0	0.0	0.0	0.0	0.0	1.0
I-IMPLICIT	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0
B-LOC	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.4	0.0	**0.6**	0.0	0.0	0.0	0.8
I-LOC	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0
B-CITATION	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0
I-CITATION	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0
O	76.4	9.6	7.2	14.0	59.6	13.6	25.0	4.0	14.0	0.0	7.8	0.0	23.4	2.0	0.6	4.6	0.6	0.0	14.4	7.0	22.4	4.0	5.0	0.0	1.0	7.0	**213585.2**

Tabelle 21: Durchschnitt der Konfusionsmatrizen aller 'COMP'-Systeme auf dem Dortmund Chat Korpus

Andere Verwechslungen entstehen meist zwischen Klassen mit ähnlichen Eigenschaften: 'NICK' ⟷ 'PER, 'GPE' ⟷ 'GEO_DE sowie 'OTH' ⟷ 'ORG'. In einem praktischen Anwendungsfall kann man davon ausgehen, dass die interne Verwechslung von solch ähnlichen Klassen ein geringes Problem darstellt. Denn ob an einer gewissen Stelle ein Nickname oder ein Personenname stand, verändert die inhaltliche Aussage des Textes kaum. Interessant ist weiterhin, dass diese Verwechslungen fast nie zwischen 'B-'- sowie 'I-Tags' geschieht. Auch wenn die Systeme die Klasse falsch voraussagen, ist die Voraussage bezüglich der BIO-Kodierung meist korrekt. Auch innerhalb der Klassen geschieht fast nie eine Verwechslung zwischen 'B-'- und 'I-'-Tags (Eine Ausnahme bildet die gelbe Markierung). Die Systeme scheinen also die Anfänge und Enden der Entitäten, welche sie detektieren, gut erkennen zu können.
Um nun auf die Fragestellung zurück zu kommen: Es wirkt Kontraintuitiv, dass die binären Metriken der Systeme so viel besser ausfallen als die positiven, wenn doch die meisten Verwechslungen der Klassen mit 'O' geschieht (und so mit in den binären Fall eingehen). Doch dies liegt an der Gewichtung der einzelnen Klassen: Die Macro-Metriken, welche oben verwendet wurden, gewichten alle Klassen unabhängig ihrer Größe gleich. Als Folge dessen fällt die Tatsache, dass die 8 'CITATION'-Tokens nicht gefunden wurden genauso stark ins Gewicht, wie wenn alle 6917 'NICK' falsch klassifiziert würden. Im oberen Szenario konnten hingegen auch keine Micro-Metriken, welche die Klassen anhand ihrer Größe gewichten, genutzt werden, da die Klasse 'O' entsprechend überwiegt. Der Grund hierfür war, dass die Systeme vor allem untereinander verglichen werden sollten. Um alleine die Verwechslungen der Systeme auf den inneren Klassen zu messen, kann man Micro-F1 Score sowie MCC erheben, wobei man die Klasse 'O' ausschließt (Tabelle 22).

System	Micro-F1	MCC
BILSTM_COMP	0,95	0,88
BILSTM_CNN_COMP	0,96	0,89
BILSTM_CNN_CRF_COMP	0,96	0,90
BILSTM_CNN_2_COMP	0,96	0,90
LCRF_COMP	0,97	0,91

Tabelle 22: Metriken der 'COMP' Systeme auf dem Dortmund Chat Korpus unter Exklusion der 'O'-Klasse

Hier ist zu erkennen, dass alle Systeme hohe Werte erreichen, wobei sich das LCRF_COMP, wenn auch nur leicht, absetzt. Die interne Verwechslung von Klassen kommt, relativ gesehen zu den absoluten Häufigkeiten, also verhältnismäßig selten vor. Dies bestätigt somit die Erkenntnisse der Konfusionsmatrix.

Die Erkenntnisse, welche auf dem Dortmund Korpus gewonnen werden konnten, spiegeln sich auch in den Konfusionsmatrizen des E-Mail Korpus wieder - daher werden diese hier nicht näher behandelt. Sie können aber im Abschnitt A des Anhangs eingesehen werden. Die Micro-Metriken unter Exklusion der 'O' Klasse sind zum Vergleich in Tabelle 23 dargestellt.

System	Micro-F1	MCC
BILSTM_COMP	0,96	0,95
BILSTM_CNN_COMP	0,97	0,97
BILSTM_CNN_CRF_COMP	0,91	0,89
BILSTM_CNN_2_COMP	0,99	0,98
LCRF_COMP	0,93	0,91

Tabelle 23: Metriken der 'COMP' Systeme auf dem E-Mail Korpus unter Exklusion der 'O'-Klasse

Interessant ist hierbei, dass die Systeme BILSTM_COMP, BILSTM_CNN_COMP sowie BILSTM_CNN_2_COMP deutlich höhere Ergebnisse als auf dem Dortmund Chat Korpus erreichen. Dies suggeriert, analog zu den Ergebnissen der Gesamtevaluation, dass ihre Trennung der Klassen auf dem E-Mail Korpus gegenüber dem Dortmund Char Korpus signifikant besser funktioniert.
Um weitere Einsichten zu erlangen, werden nun einzelne Fehler des besten Systems auf diesem Korpus, BILSTM_CNN_2_COMP, anhand konkreter Situationen betrachtet. Fehler macht das System auf dem Korpus an insgesamt 8 Stellen. Teilweise passieren diese Fehler nur auf einzelnen Tokens, teilweise auf mehreren. Der häufigste Fehler (an 3 Stellen) lässt sich auf Anreden zurück führen, welche nur einen Vornamen verwenden: An diesen Stellen annotiert das System auch den nachfolgenden Namen als 'I-PER', wie in folgender Situation:

	Hallo	Horst	kannst	du	bitte	die	Adresse
Annotation	O	B-PER	O	O	O	O	O
BILSTM_CNN_2_COMP	O	B-PER	I-PER	O	O	O	O

Tabelle 24: Fehler bei Andreden des BILSTM_CNN_2_COMP auf dem E-Mail Korpus

Dies ist wohl darauf zurück zu führen, dass der mit Abstand überwiegende Anteil an 'PER' in den Korpera aus mindestens 2 Tokens (Vor- sowie Nachnahme) besteht. Darauf folgend verursachen Adressen mit, zum Beispiel, komplexeren, mehrteiligen Städtenamen Probleme:

	Adenauer	Allee	67	72923	Hagen	am	Teutoburger	Wald
Annotation	B-LOC	I-LOC	B-NUMBER	B-NUMBER	B-GPE	I-GPE	I-GPE	I-GPE
BILSTM_CNN_2_COMP	B-LOC	I-LOC	B-NUMBER	B-NUMBER	B-GPE	I-GPE	B-GPE	I-GPE

Tabelle 25: Fehler bei Adressen des BILSTM_CNN_2_COMP auf dem E-Mail Korpus

Diese Fehler können dadurch begründet sein, dass die wenigsten 'GPE's in den Korpera aus mehr als 2 Tokens bestehen. Auch enthalten einige der Adressen folgend auf die Stadt noch ein jeweiliges Land. Daher war es für das System wohl naheliegend, diese Trennung vorzunehmen. Die restlichen der 3 Stellen bilden Einzelfälle, in welchen zum Beispiel der erste Block einer IBAN nicht erkannt wurde, während alle anderen Vorkommen von IBANs mit dem gleichen Kontext korrekt detektiert wurden. Daher werden sie hier nicht weitergehend analysiert.
Eine Klassenbasierte Analyse für KSystem gestaltet sich schwierig. Bereits öfter angesprochen wurde die unterschiedliche Struktur von Labels. Das System kennt zum Beispiel im Bezug auf Namen nur die Unterscheidungen 'TitleName' (zum Beispiel 'Herr Peters'), 'Surname' (zum Beispiel 'Schmidt') sowie 'FullName' (zum Beispiel 'Marie Bauer'), nicht aber 'Nicknames'. Daher wurden alle 'FullName' im Dortmund Chat Korpus als 'NICK' behandelt, da diese den überwiegenden Teil ausmachen, im E-Mail Korpus

hingegen als ’PER’. Auch angesprochen wurde die Problematik, dass das System einige der zu anonymisierenden Entitäten im Dortmund Chat Korpus nicht als solche kennt, wie zum Beispiel ’URL’ oder ’ROOM’. Anders ist dies beim E-Mail Korpus: Bei diesem sind keine, dem System unbekannten Entitäten enthalten, auch die Zuweisung der Labels stellt kein Problem dar. Daher lohnt es sich, ausschließlich die Matrix für den E-Mail Korpus zu betrachten. Zu beachten ist hierbei, dass die Frequenzen der Tokens durch die bereits erwähnte, unterschiedliche Tokenisierung des KSystems, anders sind

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	0	0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	18
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**93**	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	15
I-PER	0	0	2	**90**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
B-GPE	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**12**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	73	7	0	0	0	0	0	0	3
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**143**	0	0	0	0	0	0	4
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	1	0	0	4
I-LOC	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	**1**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	3	34	18	2	0	0	0	0	0	0	1	2	1	4	0	0	44	28	0	0	6	2	0	0	**1869**

Tabelle 26: Konfusionssmatrix des KSystem auf dem E-Mail Korpus

Ähnlich wie die ML-Systeme weißt auch KSystem selten Verwechslungen zwischen den ’zu anonymisierenden’-Klassen auf (magenta), erreicht aber mit einem Micro-F1-Score von 0,71 sowie einem MCC von 0,68 im Vergleich zu den anderen Systemen auch nur geringe Werte. Vier der Verwechslungen sind auf Orte (’GPE’) zurück zu führen, welche fälschlicherweise als Nachnamen (’NICK’) markiert wurden. Die untere Markierung hingegen stellt Verwechslungen zwischen ’GPE’ sowie ’LOC’ dar, welche in einem praktischen Anwendungsfall keine größeren Auswirkungen hätte. Ähnliches gilt für den Beginn von drei Nummern, welche als ’I-LOC’ markiert wurden. Diese stellen Hausnummern dar, welche durchaus auch als Teil von ’LOC’ gesehen werden können. Anders als auf dem Dortmund Chat Korpus verwechselt KSystem auch ’B-’- sowie ’I-’-Tags (gelb markiert). Meist geschieht dies dadurch, dass das System den Beginn der jeweiligen Entität einen Token zu früh beziehungsweise zu spät detektiert. Besonders stark hingegen fallen die Verwechslungen mit der ’O’-Klasse aus (blau sowie grün markiert). Das hatte sich ja auch bereits in den binären Metriken (Sektion 4.4.1) widergespiegelt. Die Anzahl der Tokens, die ’zu viel’ anonymisiert werden, basiert hauptsächlich auf der fälschlichen Detektion von Namen (Nachnamen mit ’NICK’, sowie andere Namen durch ’PER’, grün markiert). Der für die Anonymisierung kritischere Fall der zu anonymisierenden Tokens, welche nicht detektiert wurden (letzte Zeile), weißt eine deutlich höhere Anzahl an Fehlklassifikationen auf. Besonders häufig wurden Nachnahmen als Teil

von Namen (’I-PER’) sowie Orte (’GPE’, ’LOC’) und diverse Nummern (’NUMBER’) wie Bankleitzahlen oder Kontonummern nicht detektiert (siehe dazu das Beispiel aus dem letzten Abschnitt).

Zusammenfassung

Im Rahmen der Klassenorientierten Evaluation wurden zwei Zentrale Fragestellungen beantwortet. Die Frage, auf welchen Klassen sich die Systeme besonders gut beziehungsweise schlecht schlagen, führte über eine allgemeine Betrachtung der Leistungen in allen Klassen. Hier stellte sich heraus, dass die Systeme die besten Leistungen auf Klassen erbrachten, welche einen regulären Kontext über alle Vorkommen hinweg aufzuweisen haben. Auch wichtig ist die Ausprägung der unterschiedlichen Entitäten: Umso ähnlicher die Wörter einer Klasse untereinander waren, umso besser konnten die Systeme sie erkennen. Des weiteren spielt die Häufigkeit einer Klasse in den Trainingsdaten eine Rolle: Umso häufiger eine Klasse in den Trainingsdaten vertreten war, umso besser schnitten die Systeme auf ihnen ab. Auf Klassen, welche keine oder nur einen Teil dieser Attribute aufzuweisen hatten, war die Leistung bedeutend schlechter, zum Beispiel bei ’IMPLICIT’ oder ’OTH’. Dies deckt sich mit den Erkenntnissen, welche aus dem WNUT16-Korpus (vergleiche Sektion 3.2.2) gewonnen werden konnten: Auch dort war die Leistung aufgrund unregulärer Daten sowie kleinerer Klassen bedeutend schlechter als in vergleichbaren Szenarien mit regulären Daten sowie größeren Klassen.
Darauf aufbauend konnte die eigentliche Fragestellung beantwortet werden: Besonders gut über fast alle Klassen hinweg schlugen sich das LCRF_COMP sowie BILSTM_CNN_2_COMP. Ersteres zeigte einzig auf ’LOC’ im Dortmund Korpus nicht so gute Leistungen, letzteres auf ’ORG’ sowie ’PER’. Die restlichen ML-Systeme erreichten im Allgemeinen ähnliche Ergebnisse, wobei das BILSTM_CNN_CRF_COMP Schwächen auf ’GEO_DE’ zeigte, das LCRF auf ’EMAIL’ und beide BILSTMs auf ’URL’. Generell hat das zusätzliche Training auf dem E-Mail Datensatz die Leistung auf einigen Klassen auch im Dortmund Korpus verbessert (zum Beispiel ’NUMBER’), auf einigen aber auch durch den unterschiedlichen Einsatz der Labels verschlechtert (zum Beispiel ’LOC’). Des weiteren hat sich die Addition eines CNNs vor allem in der Kategorie ’URL’ als Vorteilhaft gezeigt. KSystem hingegen zeigte vor allem in den Kategorien ’EMAIL’, ’NUMBER’ sowie ’LOC’ vergleichsweise gute Leistungen, wobei diese nie über das Mittelfeld hinaus reichten. Besonders auf den anderen Klassen konnte dies zu einem Teil auf Unterschiede in den Labels zurück geführt werden. Im Allgemeinen blieb es aber mit seinen Leistungen deutlich unter den ML-Systemen.
Um zu beantworten, wodurch der Leistungsunterschied zwischen der binären- und Multiklassen-Klassifikation zustande kommt, wurde betrachtet, welche Klassen von den Systemen häufig verwechselt wurde. Dabei stellte sich her raus, dass vergleichsweise wenige solcher Verwechslungen zwischen den ’zu-anonymisierenden’ Klassen vorkommen - meist geschehen die Fehler in Zusammenspiel mit der ’O’-Klasse. Andere Verwechslungen geschahen häufig zwischen ähnlichen Klassen wie ’PER’ und ’NICK’. Solche Verwechslungen sind im Rahmen praktischer Anwendung, wie oben erläutert, aber nicht gravierend. Der Grund für den vermeintlichen Leistungsunterschied liegt hingegen in der Verwendung der ’Macro’-Metrik, welche Fehler auf kleinen Klassen (wie zum Beispiel ’CITATION’) vergleichsweise stark gewichtet. Unter Verwendung des Micro-F1-Scores sowie des MCCs konnte der Leistungsunterschied größtenteils negiert werden, wobei die Klasse ’O’ exkludiert wurde. So war es möglich, nur die Verwechslungen zwischen Klassen zu messen, welche anonymisiert werden sollen.
Für KSystem wurde eine gesonderte Analyse der Konfusionsmatrix auf dem E-Mail Korpus durchgeführt, da es dort keine Probleme mit der Zuordnung der Labels gibt. Dabei stellte sich her raus, dass auch KSystem wenige Verwechslungen zwischen den zu anonymisierenden Klassen aufweist. Die Hauptfehlerquelle sind vor allem Tokens aus den Kassen ’GPE’ sowie ’NUMBER’, welche anonymisiert werden müssten, aber von KSystem nicht detektiert werden.

4.4.3 Named Entities in der Anonymisierung

In der letzten Sektion wurde gezeigt, dass es möglich ist, NER-Systeme für die Zwecke einer Anonymisierung einzusetzen und gute Ergebnisse zu erzielen. Dabei wurde auch ausgearbeitet, dass die Systeme selbst auf Kategorien, welche normalerweise nicht zu dem Spektrum der Named Entites (NE) gehören,

gute Ergebnisse lieferten (zum Beispiel in 'URL' oder 'EMAIL'). Eine Frage gilt es aber noch zu klären: Erkennen es die Systeme korrekt, wenn NEs im Text enthalten sind, welche nicht anonymisiert werden müssen? (vergleiche Sektion 2.2.3) Im Dortmund Chat Korpus ist dies zum Beispiel bei den Namen von Prominenten der Fall, welche im Rahmen der Konversation nicht anonymisiert werden müssen. Daher wird sich diese Sektion beispielhaft mit einigen solcher Vorkommen beschäftigen um herauszufinden, welche Systeme diese Aufgabe korrekt erledigen. Das Subjekt dieser Analyse sind dabei Vorkommen aus dem Dortmund Chat Korpus - im E-Mail Korpus sind keine solche Entitäten enthalten.
Die Beispiele, anhand welcher diese Analyse vorgenommen wird, sind in zwei Tabellen aufgeteilt (Tabellen 27 sowie 28). Die erste Tabelle enthält hauptsächlich Beispiele aus der Klasse 'PER', die zweite Tabelle konzentriert sich hingegen auf Beispiele der Klassen 'GPE' sowie 'ORG'. In der linken Spalte ist der jeweilige Text, aufgeteilt in Tokens, gegeben. Darauf folgt die Spalte mit dem Label aus dem Kontext der Anonymisierung, danach das entsprechende Label, welches im Rahmen einer NER genutzt werden würde. Die Auswahl der konkreten Beispiele wurde hierbei unabhängig der Vorhersagen der Systeme, sondern anhand der Diversität ihrer Struktur getroffen. Für weitere Klassen waren keine geeigneten Beispiele vorhanden. Auch, weil es für viele von ihnen kein 'NER-Pendant' gibt.
Ein Blick in die erste Tabelle verrät, dass in den ersten drei Beispielen keines der ML-Systeme einen der Namen fälschlicherweise als einen solchen annotiert, auch die Organisation 'Hochschulrektorenkonferenz' wurde richtigerweise nicht anonymisiert. Einzig im letzten Beispiel annotieren beide Versionen des BILSTM_CNN_2 'Nicki Lauda' fälschlicherweise als Name. Interessant ist, dass beide Systeme der gleichen Bauart diese Annotation vornehmen. Doch anhand der Tatsache, dass beide BILSTM_CNN_2, wie in den vorherigen Sektionen erarbeitet, im Vergleich zu den anderen Systemen eine höheren Wert auf Recall als auf Precision legen, ist dies ein nachvollziehbares Verhalten. Insgesamt suggeriert dieses Ergebnis, dass die ML-Systeme in der Lage sind, die Differenzierung zwischen Namen (als NE), welche anonymisiert werden müssen und solchen, die nicht anonymisiert werden müssen, zu vollführen. Dass dies nicht aus einer allgemeinen Tendenz, 'PER' nicht zu anonymisieren, resultiert, hat der hohe Recall der Systeme auf dieser Klasse gezeigt. Dort haben alle ML-Systeme gut auf dieser Kategorie abgeschnitten. Anders sieht dies bei KSystem aus: Vermutlich sind Namenslisten, auf denen die verwendeten Namen enthalten sind, der Grund dafür, dass sie alle ausnahmslos anonymisiert werden (bis auf einen Fehler bei 'herr schröder'). Das System wurde auch nicht für die Anonymisierung von Texten ausgelegt, in denen 'unkritische' Namen enthalten sind. Dementsprechend kann es der hier gestellten Anforderung nicht genügen.
Die Beispiele der zweiten Tabelle enthalten vor allem Organisationen (Parteien) sowie Bundesländer als 'GPE' (in diesem Fall Bayern, für den besseren Vergleich). Sowohl Parteien als auch Bundesländer können als personenbezogene Daten gelten und müssen somit gegebenenfalls anonymisiert werden. Dies ist zum Beispiel der Fall, wenn eine Person ihre Parteizugehörigkeit preis gibt, oder jemand mit dem Bundesland seine Herkunft verrät. In diesen Beispielen jedoch beziehen sich die Informationen nicht auf private Personen und müssen daher nicht anonymisiert werden.
In beiden Fällen einer 'ORG'-Entität anonymisiert keines der Systeme die jeweilige Partei - auch KSystem nicht, wobei zu beachten ist, dass KSystem im Allgemeinen keine der 'ORG'-Entitäten in den Korpera anonymisiert hat. Das einzelne Vorkommen einer Person (Merkel) wird von den ML-Systemen richtigerweise nicht anonymisiert, während KSystem analog zu den vorherigen Beispielen eine Anonymisierung vornimmt. Im Falle der 'GPE' fällt dies anders aus: Während KSystem beide Vorkommen als 'PER' annotiert, wird nur das Zweite von allen ML-Systemen richtig erkannt. Beim ersten Vorkommen liegen sowohl BILSTM, BILSTM_CNN, BILSTM_CNN_COMP als auch BILSTM_CNN_CRF_COMP falsch. Ein Grund hierfür könnte in dem Kontext liegen: Es wird davon gesprochen, dass jemand aus Bayern kommt. Dies ist grundsätzlich eine Information, die man anonymisieren würde. Vor allem vor dem Hintergrund, dass viele ähnliche Vorkommen solcher Sätze wie zum Beispiel ('Ich bin aus Frankfurt') entsprechend in den Datensätzen annotiert sind. Dass es sich hierbei um einen Politiker handelt und die Information so nicht anonymisiert werden muss (da sie öffentlich zugänglich ist), ist wohl schwer für die Systeme zu detektieren. Im Allgemeinen konnten die Systeme auch auf diesen Beispielen die Unterscheidung zwischen NE,

die anonymisiert werden müssen und solchen, die nicht anonymisiert werden müssen, weitestgehend korrekt vornehmen. Einschränkend ist hierzu ist aber zu sagen, dass die Gesamtleistungen der Systeme auf 'ORG' sowie 'GPE' zwar, wie in den vorherigen Sektionen erarbeitet, gut waren, aber nicht so gut wie in anderen Kategorien (wie zum Beispiel 'PER'). Dementsprechend können die nicht erfolgten Anonymisierungen zumindest teilweise auch auf ein nicht ausgereiftes Verständnis der jeweiligen Klasse zurückzuführen sein.

	Anonymisierungs-Annotation	NER-Annotation	BILSTM	BILSTM_COMP	BILSTM_CNN	BILSTM_CNN_COMP	BILSTM_CNN_CRF	BILSTM_CNN_CRF_COMP	BILSTM_CNN_2	BILSTM_CNN_2_COMP	LCRF	LCRF_COMP	KSystem
glaubwürdigkeit	O	O	O	O	O	O	O	O	O	O	O	O	O
hat	O	O	O	O	O	O	O	O	O	O	O	O	O
herr	O	O	O	O	O	O	O	O	O	O	O	O	B-PER
schröder	O	B-PER	O	O	O	O	O	O	O	O	O	O	O
in	O	O	O	O	O	O	O	O	O	O	O	O	O
den	O	O	O	O	O	O	O	O	O	O	O	O	O
letzten	O	O	O	O	O	O	O	O	O	O	O	O	O
jahren	O	O	O	O	O	O	O	O	O	O	O	O	O
Klaus	O	B-PER	O	O	O	O	O	O	O	O	O	O	B-PER
Landfried	O	I-PER	O	O	O	O	O	O	O	O	O	O	B-PER
,	O	O	O	O	O	O	O	O	O	O	O	O	O
Präsident	O	O	O	O	O	O	O	O	O	O	O	O	O
der	O	O	O	O	O	O	O	O	O	O	O	O	O
Hochschulrektorenkonferenz	O	B-ORG	O	O	O	O	O	O	O	O	O	O	O
,	O	O	O	O	O	O	O	O	O	O	O	O	O
hat	O	O	O	O	O	O	O	O	O	O	O	O	O
Ja	O	O	O	O	O	O	O	O	O	O	O	O	O
-	O	O	O	O	O	O	O	O	O	O	O	O	O
und	O	O	O	O	O	O	O	O	O	O	O	O	O
Kai	O	B-PER	O	O	O	O	O	O	O	O	O	O	B-PER
Ebel	O	I-PER	O	O	O	O	O	O	O	O	O	O	I-PER
interviewt	O	O	O	O	O	O	O	O	O	O	O	O	O
Zabel	O	B-PER	O	O	O	O	O	O	O	O	O	O	B-PER
während	O	O	O	O	O	O	O	O	O	O	O	O	O
des	O	O	O	O	O	O	O	O	O	O	O	O	O
Sprints	O	O	O	O	O	O	O	O	O	O	O	O	O
...	O	O	O	O	O	O	O	O	O	O	O	O	O
und	O	O	O	O	O	O	O	O	O	O	O	O	O
Nicki	O	B-PER	O	O	O	O	O	O	B-PER	B-PER	O	O	B-PER
Lauda	O	I-PER	O	O	O	O	O	O	I-PER	I-PER	O	O	I-PER
muß	O	O	O	O	O	O	O	O	O	O	O	O	O
dann	O	O	O	O	O	O	O	O	O	O	O	O	O
bei	O	O	O	O	O	O	O	O	O	O	O	O	O
seinen	O	O	O	O	O	O	O	O	O	O	O	O	O
Interviews	O	O	O	O	O	O	O	O	O	O	O	O	O
nen	O	O	O	O	O	O	O	O	O	O	O	O	O
Radhelm	O	O	O	O	O	O	O	O	O	O	O	O	O
anziehen	O	O	O	O	O	O	O	O	O	O	O	O	O

Tabelle 27: Annotationen der verschiedenen Systeme anhand einiger Beispiele der 'PER'-Klasse des Dortmund Chat Korpus

	Anonymisierungs-Annotation	NER-Annotation	BILSTM	BILSTM_COMP	BILSTM_CNN	BILSTM_CNN_COMP	BILSTM_CNN_CRF	BILSTM_CNN_CRF_COMP	BILSTM_CNN_2	BILSTM_CNN_2_COMP	LCRF	LCRF_COMP	KSystem
wer	O	O	O	O	O	O	O	O	O	O	O	O	O
ist	O	O	O	O	O	O	O	O	O	O	O	O	O
auch	O	O	O	O	O	O	O	O	O	O	O	O	O
der	O	O	O	O	O	O	O	O	O	O	O	O	O
Meinung	O	O	O	O	O	O	O	O	O	O	O	O	O
?	O	O	O	O	O	O	O	O	O	O	O	O	O
die	O	O	O	O	O	O	O	O	O	O	O	O	O
SPD	O	B-ORG	O	O	O	O	O	O	O	O	O	O	O
äußert	O	O	O	O	O	O	O	O	O	O	O	O	O
sich	O	O	O	O	O	O	O	O	O	O	O	O	O
dagegen	O	O	O	O	O	O	O	O	O	O	O	O	O
Was	O	O	O	O	O	O	O	O	O	O	O	O	O
hat	O	O	O	O	O	O	O	O	O	O	O	O	O
denn	O	O	O	O	O	O	O	O	O	O	O	O	O
die	O	O	O	O	O	O	O	O	O	O	O	O	O
CDU	O	B-ORG	O	O	O	O	O	O	O	O	O	O	O
mit	O	O	O	O	O	O	O	O	O	O	O	O	O
Frau	O	O	O	O	O	O	O	O	O	O	O	O	B-PER
Merkel	O	B-PER	O	O	O	O	O	O	O	O	O	O	I-PER
gemacht	O	O	O	O	O	O	O	O	O	O	O	O	O
?	O	O	O	O	O	O	O	O	O	O	O	O	O
kotzt	O	O	O	O	O	O	O	O	O	O	O	O	O
mich	O	O	O	O	O	O	O	O	O	O	O	O	O
an	O	O	O	O	O	O	O	O	O	O	O	O	O
und	O	O	O	O	O	O	O	O	O	O	O	O	O
er	O	O	O	O	O	O	O	O	O	O	O	O	O
ist	O	O	O	O	O	O	O	O	O	O	O	O	O
a	O	O	O	O	O	O	O	O	O	O	O	O	O
nu	O	O	O	O	O	O	O	O	O	O	O	O	O
aus	O	O	O	O	O	O	O	O	O	O	O	O	O
bayern	O	B-GPE	B-GPE	O	B-GPE	B-GPE	O	B-GPE	O	O	O	O	B-PER
Stichwort	O	O	O	O	O	O	O	O	O	O	O	O	O
Pisa	O	O	O	O	O	O	O	O	O	O	O	O	O
,	O	O	O	O	O	O	O	O	O	O	O	O	O
Bayern	O	B-GPE	O	O	O	O	O	O	O	O	O	O	B-PER
wäre	O	O	O	O	O	O	O	O	O	O	O	O	O
im	O	O	O	O	O	O	O	O	O	O	O	O	O
Internationalen	O	O	O	O	O	O	O	O	O	O	O	O	O
vergleich	O	O	O	O	O	O	O	O	O	O	O	O	O

Tabelle 28: Annotationen der verschiedenen Systeme anhand einiger Beispiele des Dortmund Chat Korpus

4.5 Zusammenfassung

In dieser Sektion wurde anhand verschiedener Evaluationsmethodiken gezeigt, dass die ML-Systeme bei einem Einsatz auf jedem der beiden Datensätze gute Ergebnisse erreichen und dabei das Vergleichssystem aus der Industrie in fast allen Aspekten schlagen (Sektion 4.4). Aus rechtlicher Sicht besonders relevant sind hierbei die Ergebnisse der binarisierten Klassifizierung. In dieser ist es nicht relevant, ob die Systeme die korrekte Klasse der zu anonymisierenden Entität vorhersagen, sondern nur, dass sie sie korrekt als 'zu anonymisieren' klassifizieren. In diesem Szenario erreichten die besten Systeme einen Recall von 97% auf dem Chat Korpus und 99,7% auf dem E-Mail Korpus. Gleichzeitig konnten sie auch eine hohe Precision von 98% beziehungsweise 99% vorweisen. Es handelte sich hierbei um das LCRF_COMP sowie das BILSTM_CNN_2_COMP (Sektion 4.4.1). Des weiteren wurde gezeigt, dass die Konzepte der Anonymisierung auch über die sehr unterschiedliche Strukturen der beiden Datensätze hinweg übertragen werden können. Denn das zusätzliche Training des E-Mail Datensatzes erhöhte in der Regel auch die Leistung auf dem Chat Korpus.
Die Leistungen der ML-Systeme bei der Kategorisierung der Entitäten ist zweigeteilt. Auf Klassen, dessen Entitäten ähnliche Ausprägungen besitzen und in regelmäßigen Kontexten zu finden sind, ist die Leistung der Systeme besonders gut. Hier erreichen sie F1-Scores zwischen 90% und 100%. Auf Klassen hingegen, welche sehr diverse Ausprägungen bei wechselnden Kontexten aufweisen, zeigen die Systeme nur Leistungen zwischen 10% und 20% F1-Score. Das KSystem erreicht hingegen nur auf den regulären Kategorien 'EMAIL' sowie 'NUMBER' konkurrenzfähige Leistungen (Sektion 4.4.2).
Anhand einiger Beispiele wurde weiterhin herausgearbeitet, dass die ML-Systeme insbesondere in der Kategorie 'PER' - bis zu einem gewissen Grad - in der Lage sind, Named Entities, welche anonymisiert werden müssen, von solchen zu unterscheiden, die nicht anonymisiert werden müssen (Sektion 4.4.3).

5 Fazit

Diese Arbeit hat die Anonymisierung von unstrukturierten Texten deutscher Sprache behandelt. Motiviert wurde sie durch eine Problemstellung, für welche keine zufriedenstellende Lösung existiert: In den letzten Jahren hat die Analyse von großen Datenmengen unter dem Stichwort 'Big Data' stark zugenommen. Viele Unternehmen versprechen sich durch sie Vorteile, zum Beispiel im Vertrieb oder in der Optimierung interner Abläufe [84]. Doch strengere Datenschutzregelungen durch die DSGVO, welche im Frühling dieses Jahr in Kraft trat, erschweren die Verarbeitung solcher Daten (Sektion 2.1). Um Daten rechtskonform zu verarbeiten, sind effiziente, sowie zuverlässige, Möglichkeiten zur Anonymisierung notwendig. Momentane Ansätze, welche auf klassische Methoden (Sektion 3.1) setzen, zeigen unzufriedenstellende Ergebnisse, insbesondere auf unregulären Daten. Dennoch wird in diesem Bereich kaum geforscht. Als vielversprechende Alternative hat diese Arbeit den Einsatz von ML untersucht.
Als Basis für die Verwendung dieser Methoden wurden zwei Datensätze erstellt, welche jeweils einen unterschiedlichen Fokus verfolgen: Einen großen Chat Korpus, den vor allem unreguläre Daten ausmachen, sowie einen kleineren E-Mail Korpus, der die Lücken des ersten Korpus schließt: So beinhaltet dieser zum einen Texte von regulärer Struktur, zum anderen sind in ihm weitere Arten von Daten, wie zum Beispiel Bankverbindungen, enthalten. Diese fehlen im Chat Korpus, nehmen aber in der Anonymisierung, besonders für Unternehmen, eine wichtige Rolle ein. Optimal sind die Datensätze jedoch nicht, denn sie wurden beide, zu einem unterschiedlichen Grad, automatisch generiert. Es wurde bei der Erstellung zwar darauf geachtet, möglichst realitätsnahe Daten zu generieren, doch nur reale Datensätze bieten ein absolut realistisches Szenario. Solche Daten sind aus zweierlei Gründen nicht verfügbar: Zum einen bedeutet die Annotation eines entsprechend großen Datensatzes im Allgemeinen viel Arbeit, zum anderen bereitet die Veröffentlichung der dort - nach Definition - enthaltenen, personenbezogenen Daten Probleme. Daher ist die Verwendung solcher automatisch generierten Korpera eine gute Alternative. Die beiden hier vorgestellten Datensätze bilden gemeinsam, wie an den Ergebnissen zu erkennen ist, eine verlässliche Basis für ein breites Spektrum an Aufgaben in der Anonymisierung.
Verschiedene Ansätze des ML, welche den 'state-of-the-art' der NER bilden, wurden mithilfe dieser Datensätze erfolgreich auf die Aufgabe der Anonymisierung übertragen. So wurden nicht nur im regulären Kontext des E-Mail Korpus sehr gute Ergebnisse erreicht, sondern auch auf den unregulären Daten des Chat Korpus. Im Zuge dessen wurden auch Eingangs erwähnte Problematiken überwindet: Die Systeme sind - bis zu einem gewissen Grad - in der Lage, Named Entities, welche anonymisiert werden müssen, von solchen zu unterscheiden, die nicht anonymisiert werden müssen. Daraus lässt sich folgern, dass die Systeme in der Lage sind, Anonymisierung als ein Gesamtkonzept zu begreifen. Dies wird dadurch unterstützt, dass die Hinzunahme des E-Mail Korpus zu den Trainingsdaten auch die Leistungen der Systeme auf dem Chat Korpus verbesserte, obwohl dieser grundlegend unterschiedliche, textuelle Strukturen aufweist. Dies ist ein herausragendes Ergebnis.
Ihre Stärken zeigen die Systeme in besonders wichtigen Kategorien wie Personen- oder Städtenamen, sowie bei Zahlungsdaten, welche schnell zu der Identifikation einer Person führen können. Nichtsdestotrotz ist auch in diesen Bereichen eine Steigerung der Leistung möglich und nicht zuletzt notwendig, um die Systeme in der Praxis einsetzten zu können. Entsprechende Möglichkeiten dazu werden im nächsten Abschnitt diskutiert. Schwächen weisen die Systeme hingegen auf sehr diversen Kategorien, wie 'Othername' oder 'Implicit', auf. Diese Fehler sind auf der einen Seite weniger schwerwiegend, da sie im Vergleich deutlich weniger Vorkommen aufweisen und in der Regel bei einer fehlenden Anonymisierung nicht direkt zu der Identifikation einer Person führen. Auf der anderen Seite muss die Leistungen auf diesen Klassen trotzdem deutlich verbessert werden, um ein, in allen Hinsichten, verlässliches System zu erhalten. Auch dies wird daher im folgenden Abschnitt näher beleuchtet.
Das Vergleichssystem aus der Industrie, KSystem, liegt in allen Belangen hinter den ML-Systemen zurück. Insbesondere auf den unregulären Daten des Chat Korpus zeigt es schlechte Leistungen, auch die Ergebnisse im E-Mail Korpus sind nicht zufriedenstellend - es genügt den Anforderungen, die die DSGVO an eine Anonymisierung stellt, nicht. Daher werden solche Systeme zukünftig wohl keine wichtige Rol-

le mehr in der Anonymisierung einnehmen. Während die ML-Systeme hingegen zwar auch Schwächen aufweisen, bieten sie insgesamt eine bessere - solide - Leistung, insbesondere über verschiedene Textarten hinweg. Besonders gut hat die Kombination eines BILSTMs mit einem CNN abgeschlossen. Solche Ansätze werden voraussichtlich, aufgrund ihrer Flexibilität und ihren guten Leistungen, zukünftig eine wichtige Rolle in der Anonymisierung einnehmen.

5.1 Zukünftige Arbeit

Der Bereich der Anonymisierung von Textdokumenten - insbesondere von unregulären - erfährt kaum zielgerichtete Forschung. Daher sind in diesem Bereich viele Fragestellungen offen, welche es zu erforschen gilt. Im Rahmen dieser Arbeit sind dabei vier zentrale Möglichkeiten, wie man die Arbeit an diesem Thema weiterführen kann, in den Blick gerückt.

1. Eine Problematik in dieser Arbeit ist im Rahmen von Klassen mit unregelmäßigem Kontext und vielen unterschiedlichen Ausprägungen, wie zum Beispiel im Falle von 'IMPLICIT' oder 'OTH', aufgetreten. Die Leistungen der Systeme auf diesen Klassen waren nicht zufriedenstellend. Daher wäre es im Rahmen zukünftiger Arbeiten interessant zu analysieren, inwiefern man die Erkennung dieser Klassen verbessern kann. Während eine größere Menge an Trainingsdaten eine naheliegende Möglichkeit ist, die Leistung zu verbessern, sind potentiell auch andere Architekturen oder das Oversampling einzelner Klassen eine Möglichkeit.

2. Ein weiter Aspekt bildet die Optimierung der Leistung der verschiedenen Systeme. Besonders für einen alltäglichen Einsatz in Unternehmen müssen die Systeme eine kontinuierliche, verlässliche Leistung erbringen, um allen rechtlichen Ansprüchen zu genügen. Da das Ziel dieser Arbeit auf dem generellen Transfer der Systeme auf den Bereich der Anonymisierung, sowie dem grundlegenden Vergleich der Konzepte lag, wurde keine Optimierung, zum Beispiel der Hyperparameter, behandelt. Daher ergeben sich in dieser Richtung noch weitere Möglichkeiten. Da es - im Gegensatz zu der Anonymisierung - im Bereich der NER eine große Breite an Datensätze gibt, könnte man sich zum Beispiel das Konzept des Transfer-Learnings zunutze machen. So würde man zuerst die Systeme auf einem oder mehreren NER-Korpera trainieren, um sie anschließend auf Datensätze der Anonymisierung weiter zu trainieren. Durch diesen Ansatz könnte es möglich sein, ohne weitere Daten die Leistungen der Systeme zu verbessern.

3. Subjekt dieser Arbeit waren Texte deutscher Sprache. Weitere Untersuchungen wären auch der Leistung in anderen Sprachen, wie zum Beispiel Englisch, zu widmen. Doch ähnlich wie sich die Leistung aus der Englischen Sprache auf die Deutsche übertragen ließ, ist auch ein erfolgreicher Transfer in die umgekehrte Richtung zu erwarten.

4. Für eine alltägliche Anwendung in der Praxis ist es des weiteren wichtig, das jeweilige System in ein Framework einzubetten, welches aus vorhergesagten Anonymisierungen eine tatsächliche Pseudonymisierung gewinnt. Dafür ist es zum Beispiel notwendig, denselben, ersetzten Entitäten gleiche IDs zuzuweisen und das zusätzliche Dokument, welches die ursprüngliche Ausprägung aller anonymisierten Entitäten enthält, zu führen.

Sind diese Fragestellungen adressiert, werden ML-Systeme außerordentlich gute Leistungen im Bereich der Anonymisierung auf diversen Arten von Texten verschiedener Sprachen erbringen können, ohne den Einschränkungen klassischer Ansätze zu unterliegen. Als Folge dessen ist zu erwarten, dass sie zu einem Standard in der Verarbeitung textueller Daten werden.

Literatur

[1] Charu C Aggarwal and ChengXiang Zhai. *Mining text data*. Springer Science & Business Media, 2012.

[2] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4):2347–2376, 2015.

[3] Timothy Baldwin, Marie-Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135, 2015.

[4] Michael Beißwenger. Das dortmunder chat-korpus: ein annotiertes korpus zur sprachverwendung und sprachlichen variation in der deutschsprachigen chat-kommunikation. *LINSE-Linguistik Server Essen*, pages 1–13, 2013.

[5] Michael Beißwenger, Eric Ehrhardt, Axel Herold, Berlin-Brandenburgische Akademie der Wissenschaften, and Angelika Storrer. Integrating corpora of computer-mediated communication in clarin-d: Results from the curation project chatcorpus2clarin. *Bochumer Linguistische Arbeitsberichte*, page 156, 2016.

[6] Darina Benikova, Chris Biemann, and Marc Reznicek. Nosta-d named entity annotation for german: Guidelines and dataset. In *LREC*, pages 2524–2531, 2014.

[7] Darina Benikova, Seid Muhie, Yimam Prabhakaran, and Santhanam Chris Biemann. C.: Germaner: Free open german named entity recognition tool. In *In: Proc. GSCL-2015*, 2015.

[8] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. Ö'Reilly Media, Inc.", 2009.

[9] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[10] Robert Blumberg and Shaku Atre. The problem with unstructured data. *Dm Review*, 13(42-49):62, 2003.

[11] A. Boschetti and L. Massaron. *Python Data Science Essentials: Become an Efficient Data Science Practitioner by Thoroughly Understanding the Key Concepts of Python*. Community experience distilled. Packt Publishing, 2015.

[12] Sabri Boughorbel, Fethi Jarray, and Mohammed El-Anbari. Optimal classifier for imbalanced data using matthews correlation coefficient metric. *PloS one*, 12(6):e0177678, 2017.

[13] Martin Braschler and Bärbel Ripplinger. How effective is stemming and decompounding for german text retrieval? *Information Retrieval*, 7(3-4):291–316, 2004.

[14] Gavin C Cawley and Nicola LC Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(Jul):2079–2107, 2010.

[15] Tao Chen, Richard M Cullen, and Marshall Godwin. Hidden Markov model using Dirichlet process for de-identification. *Journal of biomedical informatics*, 58:S60–S66, 2015.

[16] Davide Chicco. Ten quick tips for machine learning in computational biology. *BioData Mining*, 10(1):35, Dec 2017.

[17] Jason P. C. Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *CoRR*, abs/1511.08308, 2015.

[18] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[19] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[20] Roger Clarke. Big data, big risks. *Information Systems Journal*, 26(1):77–90, 2016.

[21] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

[22] Howard B Demuth, Mark H Beale, Orlando De Jess, and Martin T Hagan. *Neural network design*. Martin Hagan, 2014.

[23] Franck Dernoncourt, Ji Young Lee, Ozlem Uzuner, and Peter Szolovits. De-identification of patient notes with recurrent neural networks. *Journal of the American Medical Informatics Association*, 24(3):596–606, 2017.

[24] Devmount. German Word Embeddings, Mai 2015.

[25] Francisco Manuel Carvalho Dias. Multilingual automated text anonymization. *Instituto Superior Técnico of Lisboa*, 2016.

[26] Duden. Definition Datenschutz, Juli 2018.

[27] Duden. Zum Umfang des deutschen Wortschatzs, Oktober 2018.

[28] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*, 2015.

[29] Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. Named entity recognition through classifier combination. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 168–171, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[30] Guangyu Ge. Natural language processing. 2007.

[31] Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12:2451–2471, 1999.

[32] Dan Gillick, Cliff Brunk, Oriol Vinyals, and Amarnag Subramanya. Multilingual language processing from bytes. *arXiv preprint arXiv:1512.00103*, 2015.

[33] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[34] Alex Graves. Supervised sequence labelling with recurrent neural networks. 2012. *ISBN 9783642212703. URL http://books. google. com/books*.

[35] David J. Hand and Robert J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning*, 45(2):171–186, Nov 2001.

[36] Christian Hänig, Stefan Thomas, and Stefan Bordag. Modular classifier ensemble architecture for named entity recognition on low resource systems. 2014.

[37] Christian Hänig, Stefan Thomas, and Stefan Bordag. Modular classifier ensemble architecture for named entity recognition on low resource systems. 2014.

[38] Ibrahim Abaker Targio Hashem, Ibrar Yaqoob, Nor Badrul Anuar, Salimah Mokhtar, Abdullah Gani, and Samee Ullah Khan. The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, 47:98 – 115, 2015.

[39] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[40] Isohrab. German-NER, November 2017.

[41] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[42] Dan Jurafsky and James H Martin. *Speech and language processing*, volume 3. Pearson London, 2014.

[43] Giuseppe Jurman, Samantha Riccadonna, and Cesare Furlanello. A comparison of mcc and cen error measures in multi-class prediction. *PLOS ONE*, 7(8):1–8, 08 2012.

[44] Anders Krogh, BjoÈrn Larsson, Gunnar Von Heijne, and Erik LL Sonnhammer. Predicting transmembrane protein topology with a hidden markov model: application to complete genomes. *Journal of molecular biology*, 305(3):567–580, 2001.

[45] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

[46] Thomas CW Landgrebe and Robert PW Duin. Approximating the multiclass roc by pairwise analysis. *Pattern recognition letters*, 28(13):1747–1758, 2007.

[47] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[48] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[49] Honglak Lee, Peter Pham, Yan Largman, and Andrew Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1096–1104. Curran Associates, Inc., 2009.

[50] Wolfgang Lezius, Reinhard Rapp, and Manfred Wettler. A freely available morphological analyzer, disambiguator and context sensitive lemmatizer for german. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 2*, COLING '98, pages 743–748, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics.

[51] Zengjian Liu, Yangxin Chen, Buzhou Tang, Xiaolong Wang, Qingcai Chen, Haodi Li, Jingfeng Wang, Qiwen Deng, and Suisong Zhu. Automatic de-identification of electronic medical records using token-level and character-level conditional random fields. *Journal of Biomedical Informatics*, 58:S47 – S52, 2015. Proceedings of the 2014 i2b2/UTHealth Shared-Tasks and Workshop on Challenges in Natural Language Processing for Clinical Data.

[52] Zengjian Liu, Buzhou Tang, Xiaolong Wang, and Qingcai Chen. De-identification of clinical notes via recurrent neural network and conditional random field. *Journal of biomedical informatics*, 75:S34–S42, 2017.

[53] Harald Lüngen, Michael Beißwenger, Laura Herzberg, and Cathrin Pichler. *Anonymisation of the Dortmund Chat Corpus 2.1*. Institut für Deutsche Sprache, Bibliothek, 2017.

[54] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

[55] Matjaz Zwitter, Milan Sokli. *Breast Cancer Data Set*. Institute of Oncology University Medical Center Ljubljana, Yugoslavia, 1988.

[56] B.W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442 – 451, 1975.

[57] José Ramon Méndez, Eva Lorenzo Iglesias, Florentino Fdez-Riverola, Fernando Díaz, and Juan M Corchado. Tokenising, stemming and stopword removal on anti-spam filtering domain. In *Conference of the Spanish Association for Artificial Intelligence*, pages 449–458. Springer, 2005.

[58] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[59] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[60] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[61] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, 2013.

[62] Ishna Neamatullah, Margaret M Douglass, H Lehman Li-wei, Andrew Reisner, Mauricio Villarroel, William J Long, Peter Szolovits, George B Moody, Roger G Mark, and Gari D Clifford. Automated de-identification of free-text medical records. *BMC medical informatics and decision making*, 8(1):32, 2008.

[63] Ziad Obermeyer and Ezekiel J Emanuel. Predicting the future—big data, machine learning, and clinical medicine. *The New England journal of medicine*, 375(13):1216, 2016.

[64] Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon infused phrase embeddings for named entity resolution. *arXiv preprint arXiv:1404.5367*, 2014.

[65] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[66] Andreas Pfitzmann and Marit Hansen. Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management-a consolidated proposal for terminology. *Version v0*, 31:15, 2008.

[67] Klaus Pommerening. *Datenschutz und Datensicherheit*. BI-Wiss.-Verlag, 1991.

[68] Nils Reimers, Judith Eckle-Kohler, Carsten Schnober, Jungi Kim, and Iryna Gurevych. Germeval-2014: Nested named entity recognition with neural networks. 2014.

[69] G Rücker, T Schimek-Jasch, and U Nestle. Measuring inter-observer agreement in contour delineation of medical imaging in a dummy run using fleiss' kappa. *Methods of information in medicine*, 51(06):489–494, 2012.

[70] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*, 2014.

[71] Cicero Nogueira dos Santos and Victor Guimaraes. Boosting named entity recognition with neural character embeddings. *arXiv preprint arXiv:1505.05008*, 2015.

[72] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *Artificial Neural Networks–ICANN 2010*, pages 92–101. Springer, 2010.

[73] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85 – 117, 2015.

[74] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[75] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[76] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[77] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[78] Benjamin Strauss, Bethany Toma, Alan Ritter, Marie-Catherine de Marneffe, and Wei Xu. Results of the wnut16 named entity recognition shared task. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 138–144, 2016.

[79] Amber Stubbs, Christopher Kotfila, and Özlem Uzuner. Automated systems for the de-identification of longitudinal clinical narratives: Overview of 2014 i2b2/uthealth shared task track 1. *Journal of Biomedical Informatics*, 58:S11 – S19, 2015. Proceedings of the 2014 i2b2/UTHealth Shared-Tasks and Workshop on Challenges in Natural Language Processing for Clinical Data.

[80] Charles Sutton, Andrew McCallum, et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373, 2012.

[81] Marie-Theres Tinnefeld, Benedikt Buchner, Thomas Petri, and Hans-Joachim Hof. *Einführung in das Datenschutzrecht: Datenschutz und Informationsfreiheit in europäischer Sicht*. Walter de Gruyter GmbH & Co KG, 2017.

[82] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada, 2003.

[83] Özlem Uzuner, Yuan Luo, and Peter Szolovits. Evaluating the state-of-the-art in automatic de-identification. *Journal of the American Medical Informatics Association*, 14(5):550–563, 2007.

[84] Bernd Wachter. Big data-anwendungen in der marktforschung. In *Big Data*, pages 17–25. Springer, 2018.

[85] Gary M Weiss and Foster Provost. The effect of class distribution on classifier learning: an empirical study. *Rutgers Univ*, 2001.

[86] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: Theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1192–1199, New York, NY, USA, 2008. ACM.

Anhang

A Durchschnitt der Konfusionssmatrizen aller 'COMP'-Systeme auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0.0**	0.0	1.8	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-NICK	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-PER	0.0	0.0	**91.0**	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	1.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-PER	0.0	0.0	0.2	**118.4**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	1.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6
B-GPE	0.0	0.0	0.0	0.0	**23.8**	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6	0.0	0.0	0.0	0.0
I-GPE	0.0	0.0	0.0	0.0	0.4	**3.8**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-OTH	0.0	0.0	0.4	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-OTH	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-GEO_DE	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-GEO_DE	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-URL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-URL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-ORG	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.8**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-ORG	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**1.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-EMAIL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**2.6**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-EMAIL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**12.6**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-ROOM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-ROOM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B-NUMBER	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**117.8**	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I-NUMBER	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6	**176.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.4
B-IMPLICIT	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0
I-IMPLICIT	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0	0.0	0.0	0.0
B-LOC	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**9.2**	0.0	0.0	0.0	0.0
I-LOC	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	**5.0**	0.0	0.0	0.0
B-CITATION	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0	0.0
I-CITATION	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.0**	0.0
O	0.0	0.0	4.2	5.8	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.8	0.0	2.4	0.0	0.0	1.8	0.8	0.0	0.0	0.2	0.0	0.0	0.0	**1907.4**

Tabelle 29: Durchschnitt der Konfusionsmatrizen aller 'COMP'-Systeme auf dem E-Mail Korpus

B Ergebnisse des Trainings ausschließlich auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**93**	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**122**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
B-GPE	0	0	0	0	**21**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	1	0	0	0	0	0	0	0	0	3	**13**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**117**	2	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	**171**	0	0	0	0	0	0	1
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**10**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	3	1	3	3	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	**1875**

Tabelle 30: Konfusionssmatrix des BILSTM_CNN_2_ONLY_EMAIL auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.99	0.75	0.99	0.72	0.99	0.72	0.99	0.97

Tabelle 31: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_2_ONLY_EMAIL auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.67	0.96	0.64	0.96	0.65	0.96	0.95

Tabelle 32: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_2_ONLY_EMAIL auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.99	0.99	0.98	0.00	0.99	0.98

Tabelle 33: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_2_ONLY_EMAIL auf dem E-Mail Korpus

C Ergebnisse BILSTM

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6865**	0	7	2	3	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	118
I-NICK	0	**810**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
B-PER	4	0	**106**	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20
I-PER	0	0	0	**97**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20
B-GPE	1	0	0	0	**121**	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	25
I-GPE	0	0	0	0	0	**41**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
B-OTH	0	0	0	0	0	0	**9**	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	7
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	10
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**18**	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	3	0	0	0	0	0	**44**	0	0	0	0	0	0	0	0	0	1	0	0	0	18
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**40**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**346**	0	0	0	0	0	0	0	0	0	1
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**8**	1	0	0	0	0	0	0	10
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**11**	0	0	0	0	0	0	3
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	3
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	138	32	11	28	77	18	26	4	15	0	30	0	18	2	1	6	2	0	15	7	23	4	5	0	1	7	**213635**

Tabelle 34: Konfusionssmatrix des BILSTM auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.56	1.00	0.44	1.00	0.48	1.00	0.96

Tabelle 35: Ergebnisse für den Fall der Klassfikation des BILSTM auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.94	0.63	0.94	0.40	0.94	0.46	0.94	0.86

Tabelle 36: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.97	0.95	0.00	0.96	0.96

Tabelle 37: Ergebnisse für den Fall der binären Klassfikation des BILSTM auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	11	7	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**46**	22	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**67**	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**16**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**8**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**27**	2	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	**13**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	1	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	41	32	9	0	0	0	0	0	0	0	2	2	0	7	0	0	89	163	0	0	8	4	0	0	**1916**

Tabelle 38: Konfusionssmatrix des BILSTM auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.84	0.60	0.84	0.34	0.84	0.40	0.84	0.53

Tabelle 39: Ergebnisse für den Fall der Klassfikation des BILSTM auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.31	0.54	0.31	0.27	0.31	0.34	0.31	0.36

Tabelle 40: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.86	1.00	0.40	0.00	0.57	0.58

Tabelle 41: Ergebnisse für den Fall der binären Klassfikation des BILSTM auf dem E-Mail Korpus

D Ergebnisse BILSTM_COMP

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6868**	0	4	3	3	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	134
I-NICK	0	**824**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	29
B-PER	9	0	**111**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20
I-PER	0	0	0	**106**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22
B-GPE	0	0	0	0	**137**	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	27
I-GPE	0	0	0	0	1	**38**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
B-OTH	0	0	0	0	0	0	**11**	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	6
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	4
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**23**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	3	0	0	0	0	0	**45**	0	0	0	0	0	0	0	0	0	1	0	0	0	15
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	1
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**40**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	3
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**8**	1	0	0	0	0	0	0	13
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**10**	0	0	0	0	0	0	22
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	3
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	**1**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	131	18	9	18	60	21	23	4	16	0	28	0	17	2	1	6	1	0	14	9	23	4	5	0	1	7	**213582**

Tabelle 42: Konfusionssmatrix des BILSTM_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.52	1.00	0.44	1.00	0.46	1.00	0.96

Tabelle 43: Ergebnisse für den Fall der Klassfikation des BILSTM_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.95	0.61	0.95	0.40	0.95	0.46	0.95	0.88

Tabelle 44: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_COMP auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.97	0.95	0.00	0.96	0.96

Tabelle 45: Ergebnisse für den Fall der binären Klassfikation des BILSTM_COMP auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**92**	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	1	**122**	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**24**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
I-GPE	0	0	0	0	1	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**11**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**119**	0	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**178**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	2	6	0	0	0	0	0	0	0	0	1	1	0	4	0	0	1	0	0	0	0	0	0	0	**1916**

Tabelle 46: Konfusionssmatrix des BILSTM_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.99	0.91	0.99	0.79	0.99	0.83	0.99	0.97

Tabelle 47: Ergebnisse für den Fall der Klassfikation des BILSTM_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.84	0.96	0.72	0.96	0.76	0.96	0.95

Tabelle 48: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_COMP auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.99	1.00	0.97	0.00	0.99	0.98

Tabelle 49: Ergebnisse für den Fall der binären Klassfikation des BILSTM_COMP auf dem E-Mail Korpus

E Ergebnisse BILSTM_CNN

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6928**	0	8	3	3	0	0	0	0	0	2	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	121
I-NICK	0	**833**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	62
B-PER	2	0	**100**	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13
I-PER	3	0	0	**101**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17
B-GPE	1	0	0	0	**112**	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19
I-GPE	0	0	0	0	0	**43**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
B-OTH	1	0	0	0	0	0	**11**	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	4
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**42**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	2	0	0	0	0	0	**44**	0	0	0	0	0	0	0	0	0	1	0	0	0	15
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	1
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**40**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	2
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**7**	0	0	0	0	0	0	0	2
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**14**	0	0	0	0	0	0	5
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	1
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	72	9	16	22	86	16	25	4	14	0	7	0	19	2	0	5	1	0	16	6	23	4	4	0	1	7	**213620**

Tabelle 50: Konfusionssmatrix des BILSTM_CNN auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.58	1.00	0.46	1.00	0.50	1.00	0.96

Tabelle 51: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.63	0.96	0.43	0.96	0.49	0.96	0.89

Tabelle 52: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.97	0.96	0.00	0.97	0.96

Tabelle 53: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	10	9	0	0	0	0	0	0	0	0	0	0	0	1	0	0	9	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**41**	24	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**64**	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**9**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	1	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	1	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**8**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**32**	5	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	**10**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	47	31	13	0	0	0	0	0	0	0	2	2	0	7	0	0	78	163	0	0	8	5	0	0	**1916**

Tabelle 54: Konfusionssmatrix des BILSTM_CNN auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.83	0.63	0.83	0.34	0.83	0.40	0.83	0.51

Tabelle 55: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.29	0.57	0.29	0.27	0.29	0.34	0.29	0.33

Tabelle 56: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.86	1.00	0.40	0.00	0.57	0.58

Tabelle 57: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN auf dem E-Mail Korpus

F Ergebnisse BILSTM_CNN_COMP

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6915**	0	2	2	3	0	1	0	0	0	3	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	121
I-NICK	0	**835**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	74
B-PER	7	0	**113**	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19
I-PER	1	0	0	**106**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15
B-GPE	1	0	0	0	**120**	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22
I-GPE	0	0	0	0	1	**41**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7
B-OTH	0	0	0	0	0	0	**10**	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	3
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	1	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	5
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**46**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	3	0	0	0	0	0	**45**	0	0	0	0	0	0	0	0	0	1	0	0	0	17
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**40**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	4
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**7**	0	0	0	0	0	0	0	5
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**12**	0	0	0	0	0	0	11
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	84	7	9	18	76	18	24	4	15	0	2	0	18	2	0	5	1	0	15	8	22	4	5	0	1	7	**213586**

Tabelle 58: Konfusionssmatrix des BILSTM_CNN_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.55	1.00	0.46	1.00	0.48	1.00	0.96

Tabelle 59: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.59	0.96	0.42	0.96	0.47	0.96	0.89

Tabelle 60: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_COMP auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.97	0.96	0.00	0.96	0.96

Tabelle 61: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_COMP auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**93**	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**125**	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**25**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
I-GPE	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	2	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**15**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**120**	0	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**178**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	1	3	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	**1916**

Tabelle 62: Konfusionssmatrix des BILSTM_CNN_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.99	0.86	0.99	0.76	0.99	0.80	0.99	0.98

Tabelle 63: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.97	0.80	0.97	0.69	0.97	0.73	0.97	0.97

Tabelle 64: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_COMP auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	1.00	0.99	0.00	0.99	0.99

Tabelle 65: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_COMP auf dem E-Mail Korpus

G Ergebnisse BILSTM_CNN_CRF

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6922**	0	7	3	1	0	0	0	0	0	3	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	118
I-NICK	0	**837**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	89
B-PER	8	0	**104**	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24
I-PER	3	0	1	**105**	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	27
B-GPE	0	0	0	0	**125**	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	18
I-GPE	0	0	0	0	1	**33**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
B-OTH	0	0	0	0	0	0	**11**	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	8
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	8
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**42**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	2	0	0	0	0	0	**44**	0	0	0	0	0	0	0	0	0	1	0	0	0	13
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**40**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	3
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	0	0	0	0	0	0	0	6
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**14**	0	0	0	0	0	0	13
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	3
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	1
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	73	5	12	18	74	26	25	4	13	0	6	0	19	2	0	5	0	0	14	6	22	4	4	0	1	7	**213557**

Tabelle 66: Konfusionssmatrix des BILSTM_CNN_CRF auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.53	1.00	0.45	1.00	0.48	1.00	0.96

Tabelle 67: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_CRF auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.61	0.96	0.42	0.96	0.48	0.96	0.89

Tabelle 68: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_CRF auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.96	0.96	0.00	0.96	0.96

Tabelle 69: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_CRF auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	11	15	0	0	0	0	0	0	0	0	0	0	0	1	0	0	5	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**45**	24	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**68**	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**16**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**8**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**42**	0	0	0	1	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	**14**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	1	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	42	21	9	1	0	0	0	0	0	0	2	2	0	7	0	0	68	164	0	0	6	4	0	0	**1916**

Tabelle 70: Konfusionssmatrix des BILSTM_CNN_CRF auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.84	0.63	0.84	0.36	0.84	0.44	0.84	0.55

Tabelle 71: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_CRF auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.34	0.57	0.34	0.29	0.34	0.37	0.34	0.37

Tabelle 72: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_CRF auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.87	1.00	0.45	0.00	0.62	0.62

Tabelle 73: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_CRF auf dem E-Mail Korpus

H Ergebnisse BILSTM_CNN_CRF_COMP

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6926**	0	9	5	3	0	3	0	0	0	2	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	140
I-NICK	0	**838**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	76
B-PER	10	0	**107**	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26
I-PER	1	0	1	**108**	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	28
B-GPE	0	0	0	0	**135**	0	1	0	3	0	0	0	1	0	0	0	0	0	0	0	0	0	3	0	0	0	40
I-GPE	0	0	0	0	2	**53**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	31
B-OTH	0	0	0	0	0	0	**7**	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	4
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**43**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	5	0	1	0	0	0	**42**	0	0	0	0	0	0	0	0	0	1	0	0	0	15
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	1	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**46**	0	0	0	0	0	0	0	0	0	0	2
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	4
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	0	0	0	1	0	0	0	17
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	**14**	0	0	0	0	0	0	24
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	**1**	0	0	0	1
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	65	4	7	13	61	6	23	4	15	0	6	0	19	2	0	0	0	0	12	6	23	4	3	0	1	7	**213481**

Tabelle 74: Konfusionssmatrix des BILSTM_CNN_CRF_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.46	1.00	0.47	1.00	0.46	1.00	0.96

Tabelle 75: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_CRF_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.52	0.96	0.43	0.96	0.46	0.96	0.90

Tabelle 76: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_CRF_COMP auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.96	0.97	0.00	0.96	0.96

Tabelle 77: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_CRF_COMP auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	3	5	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**83**	7	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**105**	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**24**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
I-GPE	0	0	0	0	1	**4**	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**12**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**118**	0	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**178**	0	0	0	0	0	0	1
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	**5**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	12	11	0	0	0	0	0	0	0	0	1	1	0	3	0	0	0	0	0	0	0	0	0	0	**1915**

Tabelle 78: Konfusionssmatrix des BILSTM_CNN_CRF_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.98	0.88	0.98	0.76	0.98	0.80	0.98	0.95

Tabelle 79: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_CRF_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.91	0.81	0.91	0.69	0.91	0.73	0.91	0.89

Tabelle 80: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_CRF_COMP auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.99	1.00	0.95	0.00	0.98	0.97

Tabelle 81: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_CRF_COMP auf dem E-Mail Korpus

I Ergebnisse BILSTM_CNN_2

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6908**	0	10	1	5	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	93
I-NICK	0	**834**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	37
B-PER	18	0	**102**	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	31
I-PER	17	0	0	**108**	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36
B-GPE	3	0	0	0	**138**	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	32
I-GPE	0	0	0	0	2	**49**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21
B-OTH	0	0	0	0	0	0	**9**	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	3
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**50**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	2	0	0	0	0	0	**45**	0	0	0	0	0	0	0	0	0	0	0	0	0	6
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	1
B-EMAIL	1	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**46**	0	0	0	0	0	0	0	0	0	0	7
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	2
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	2	0	0	0	0	0	0	12
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**14**	0	0	0	0	0	0	13
B-IMPLICIT	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	12
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	4
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	61	8	12	17	52	10	27	4	14	0	1	0	22	2	0	0	1	0	15	5	21	4	6	0	1	7	**213568**

Tabelle 82: Konfusionssmatrix des BILSTM_CNN_2 auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.59	1.00	0.58	1.00	0.58	1.00	0.96

Tabelle 83: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_2 auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.71	0.96	0.54	0.96	0.59	0.96	0.90

Tabelle 84: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_2 auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.97	0.97	0.00	0.97	0.97

Tabelle 85: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_2 auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**51**	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**72**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**10**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	1	3	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	0	2
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	1	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	2	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**13**	0	0	0	0	0	0	0	0	0	0	2
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**44**	2	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	**14**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	39	28	10	0	0	0	0	0	0	0	2	2	2	0	0	0	72	156	0	0	9	2	0	0	**1874**

Tabelle 86: Konfusionssmatrix des BILSTM_CNN_2 auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.85	0.61	0.85	0.39	0.85	0.40	0.85	0.57

Tabelle 87: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_2 auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.37	0.56	0.37	0.32	0.37	0.34	0.37	0.41

Tabelle 88: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_2 auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.87	0.98	0.44	0.00	0.61	0.61

Tabelle 89: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_2 auf dem E-Mail Korpus

J Ergebnisse BILSTM_CNN_2_COMP

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6914**	0	4	1	6	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	84
I-NICK	0	**840**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	63
B-PER	17	0	**109**	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	42
I-PER	9	0	1	**112**	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	48
B-GPE	1	0	0	0	**139**	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	42
I-GPE	0	0	0	0	2	**53**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18
B-OTH	0	0	0	0	0	0	**5**	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	2	0	1	0	0	0	**7**	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	7
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**50**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	3	0	0	0	0	0	**29**	0	0	0	0	0	0	0	0	0	0	0	0	0	4
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	1	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	1
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**46**	0	0	0	0	0	0	0	0	0	0	1
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	1
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	**10**	3	0	0	0	0	0	0	12
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**13**	0	0	0	0	0	0	12
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	1
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	**1**	0	0	0	3
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	66	2	8	13	52	6	30	4	12	0	1	0	39	2	0	0	0	0	14	5	21	4	6	0	1	7	**213545**

Tabelle 90: Konfusionssmatrix des BILSTM_CNN_2_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.61	1.00	0.56	1.00	0.56	1.00	0.96

Tabelle 91: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_2_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.70	0.96	0.52	0.96	0.57	0.96	0.90

Tabelle 92: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_2_COMP auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.96	0.97	0.00	0.97	0.96

Tabelle 93: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_2_COMP auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**96**	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**124**	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	3
B-GPE	0	0	0	0	**23**	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	1	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**13**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**118**	1	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**172**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**10**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	**1875**

Tabelle 94: Konfusionssmatrix des BILSTM_CNN_2_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.88	1.00	0.86	1.00	0.87	1.00	0.99

Tabelle 95: Ergebnisse für den Fall der Klassfikation des BILSTM_CNN_2_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.99	0.80	0.99	0.78	0.99	0.79	0.99	0.98

Tabelle 96: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des BILSTM_CNN_2_COMP auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.99	1.00	0.00	1.00	0.99

Tabelle 97: Ergebnisse für den Fall der binären Klassfikation des BILSTM_CNN_2_COMP auf dem E-Mail Korpus

K Ergebnisse LCRF

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6951**	0	3	1	2	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	58
I-NICK	0	**824**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
B-PER	4	0	**113**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18
I-PER	3	0	0	**113**	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20
B-GPE	0	0	0	0	**146**	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19
I-GPE	0	0	0	0	1	**42**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
B-OTH	0	0	0	0	0	0	**11**	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	2
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**9**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**49**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	2	0	0	0	0	0	**46**	0	0	0	0	0	0	0	0	0	1	0	0	0	4
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**16**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	3
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	1	0	0	0	0	0	0	2
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	50	18	8	13	52	17	25	4	11	0	2	0	20	2	3	30	1	0	19	11	23	4	7	0	1	7	**213748**

Tabelle 98: Konfusionssmatrix des LCRF auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.71	1.00	0.51	1.00	0.56	1.00	0.97

Tabelle 99: Ergebnisse für den Fall der Klassfikation des LCRF auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.96	0.71	0.96	0.46	0.96	0.53	0.96	0.90

Tabelle 100: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des LCRF auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.98	0.96	0.00	0.97	0.97

Tabelle 101: Ergebnisse für den Fall der binären Klassfikation des LCRF auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	10	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**49**	26	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**75**	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**12**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	39	23	12	1	0	0	0	0	0	0	2	2	1	12	0	0	121	178	0	0	9	5	0	0	**1916**

Tabelle 102: Konfusionssmatrix des LCRF auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.82	0.53	0.82	0.28	0.82	0.33	0.82	0.47

Tabelle 103: Ergebnisse für den Fall der Klassfikation des LCRF auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.24	0.47	0.24	0.21	0.24	0.27	0.24	0.31

Tabelle 104: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des LCRF auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.84	1.00	0.32	0.00	0.48	0.51

Tabelle 105: Ergebnisse für den Fall der binären Klassfikation des LCRF auf dem E-Mail Korpus

L Ergebnisse LCRF_COMP

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**6965**	0	5	3	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	68
I-NICK	0	**825**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
B-PER	4	0	**116**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17
I-PER	3	0	0	**116**	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20
B-GPE	0	0	0	0	**149**	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	21
I-GPE	0	0	0	0	1	**42**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
B-OTH	0	0	0	0	0	0	**11**	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**8**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**49**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	2	0	0	0	0	0	**42**	0	0	0	0	0	0	0	0	0	1	0	0	0	4
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**34**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**347**	0	0	0	0	0	0	0	0	0	3
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	1	0	0	1	0	0	0	5
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**13**	0	0	0	0	0	0	2
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	1
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	36	17	3	8	49	17	25	4	12	0	2	0	24	2	2	12	1	0	17	7	23	4	6	0	1	7	**213732**

Tabelle 106: Konfusionssmatrix des LCRF_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
1.00	0.68	1.00	0.55	1.00	0.59	1.00	0.97

Tabelle 107: Ergebnisse für den Fall der Klassfikation des LCRF_COMP auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.97	0.71	0.97	0.50	0.97	0.56	0.97	0.91

Tabelle 108: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des LCRF_COMP auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
1.00	0.98	0.97	0.00	0.98	0.97

Tabelle 109: Ergebnisse für den Fall der binären Klassfikation des LCRF_COMP auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**91**	3	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
I-PER	0	0	0	**116**	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
B-GPE	0	0	0	0	**23**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**12**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**114**	0	0	0	0	0	0	0	0
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**174**	0	0	0	0	0	0	1
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**9**	0	0	0	0
I-LOC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	6	9	2	0	0	0	0	0	0	0	1	1	0	4	0	0	6	4	0	0	1	0	0	0	**1915**

Tabelle 110: Konfusionssmatrix des LCRF_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.98	0.92	0.98	0.77	0.98	0.82	0.98	0.96

Tabelle 111: Ergebnisse für den Fall der Klassfikation des LCRF_COMP auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.93	0.85	0.93	0.70	0.93	0.75	0.93	0.91

Tabelle 112: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des LCRF_COMP auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.99	1.00	0.94	0.00	0.97	0.96

Tabelle 113: Ergebnisse für den Fall der binären Klassfikation des LCRF_COMP auf dem E-Mail Korpus

M Ergebnisse KSystem

Ergebnisse auf dem Dortmund Chat Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**4654**	0	103	7	26	2	5	0	2	0	9	0	7	0	0	0	58	0	0	1	0	0	0	0	0	0	5958
I-NICK	117	**0**	4	103	1	0	1	0	2	0	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	502
B-PER	3	0	**0**	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	140
I-PER	17	0	6	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	49
B-GPE	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	140
I-GPE	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**40**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	89	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	**10**	4	0	0	0	0	0	0	445
I-NUMBER	4	0	1	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	3	**14**	0	0	0	0	0	0	125
B-IMPLICIT	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	**0**	0	1	0	0	0	292
I-IMPLICIT	1	0	5	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	**0**	0	0	0	0	5
B-LOC	243	0	0	0	31	4	2	0	3	0	0	0	1	0	0	0	0	0	0	0	0	0	**1**	0	0	0	1390
I-LOC	4	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	211
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	1876	842	5	16	141	53	30	4	16	0	871	0	64	8	1	6	290	0	13	4	24	4	7	0	1	9	**204645**

Tabelle 114: Konfusionssmatrix des KSystem auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.94	0.15	0.94	0.18	0.94	0.15	0.94	0.40

Tabelle 115: Ergebnisse für den Fall der Klassfikation des KSystem auf dem Dortmund Chat Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.48	0.15	0.48	0.14	0.48	0.14	0.48	0.24

Tabelle 116: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des KSystem auf dem Dortmund Chat Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.94	0.38	0.57	0.04	0.45	0.43

Tabelle 117: Ergebnisse für den Fall der binären Klassfikation des KSystem auf dem Dortmund Chat Korpus

Ergebnisse auf dem E-Mail Korpus

	B-NICK	I-NICK	B-PER	I-PER	B-GPE	I-GPE	B-OTH	I-OTH	B-GEO_DE	I-GEO_DE	B-URL	I-URL	B-ORG	I-ORG	B-EMAIL	I-EMAIL	B-ROOM	I-ROOM	B-NUMBER	I-NUMBER	B-IMPLICIT	I-IMPLICIT	B-LOC	I-LOC	B-CITATION	I-CITATION	O
B-NICK	**0**	0	0	0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	18
I-NICK	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-PER	0	0	**93**	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	15
I-PER	0	0	2	**90**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
B-GPE	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GPE	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-OTH	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-OTH	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-GEO_DE	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-GEO_DE	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-URL	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-URL	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B-ORG	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I-ORG	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
B-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0	0
I-EMAIL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**12**	0	0	0	0	0	0	0	0	0	0	0
B-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
I-ROOM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
B-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**73**	7	0	0	0	0	0	0	3
I-NUMBER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**143**	0	0	0	0	0	0	4
B-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
I-IMPLICIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
B-LOC	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	1	0	0	4
I-LOC	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	**1**	0	0	0
B-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
I-CITATION	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
O	0	0	3	34	18	2	0	0	0	0	0	0	1	2	1	4	0	0	44	28	0	0	6	2	0	0	**1869**

Tabelle 118: Konfusionssmatrix des KSystem auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.91	0.57	0.91	0.43	0.91	0.46	0.91	0.77

Tabelle 119: Ergebnisse für den Fall der Klassfikation des KSystem auf dem E-Mail Korpus

ACC	Macro_P	Micro_P	Macro_R	Micro_R	Macro_F1	Micro_F1	MCC
0.71	0.53	0.71	0.36	0.71	0.41	0.71	0.68

Tabelle 120: Ergebnisse für den Fall der Klassfikation ohne die Klasse 'O' des KSystem auf dem E-Mail Korpus

Accuracy	Precision	Recall / TPR	FPR	F1-Score	MCC
0.92	0.90	0.76	0.03	0.82	0.77

Tabelle 121: Ergebnisse für den Fall der binären Klassfikation des KSystem auf dem E-Mail Korpus